Towards Designing Inherently Interpretable Deep Neural Networks for Image Classification

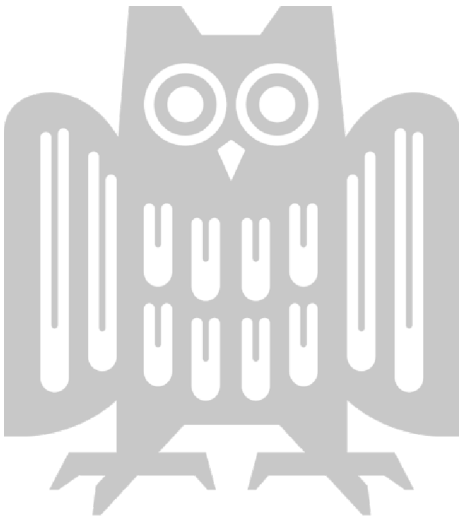
Moritz Böhle

A dissertation submitted towards the degree

Doctor of Engineering Science (Dr.-Ing.)

of the Faculty of Mathematics and Computer Science
of Saarland University

Saarbrücken, 2024



Date of Colloquium:	Friday, May 3, 2024
Dean of the Faculty:	Prof. Dr. Roland Speicher
Chair of the Committee:	Prof. Dr. Peter Ochs
Reviewers:	Prof. Dr. Bernt Schiele
	Prof. Dr. Zeynep Akata
	Dr. Wieland Brendel
Academic Assistant:	Dr. Jonas Fischer

ABSTRACT

Over the last decade, Deep Neural Networks (DNNs) have proven successful in a wide range of applications and hold the promise to have a positive impact on our lives, especially in high-stakes applications. For example, given their outstanding performance—by now regularly outperforming humans—DNNs could make state-of-the-art medical diagnostics more easily accessible to many and lessen the strain of often overworked medical professionals. That said, it is of course exactly those high-stakes situations in which a *wrong* decision can be disastrous, potentially putting human lives at risk. Especially in such settings it is therefore imperative that we can understand and obtain an explanation for a model's 'decision'.

This thesis studies this problem for image classification models from three directions. **First**, we evaluate methods that explain DNNs in a post-hoc fashion and highlight promises and shortcomings of existing approaches. In particular, we study a popular importance attribution technique to explain a model trained to identify brain scans of patients suffering from Alzheimer's disease (AD), and find it to correlate with known biomarkers of AD. Unfortunately, however, we do not know for certain which patterns in the input signals a given model is using to classify its inputs. To address this, we additionally design a novel evaluation scheme for explanation methods. Specifically, in this scheme, we control which input regions the model was certainly *not* using, which allows us to detect instances in which explanation methods are provably not model-faithful, i.e., they do not adequately represent the underlying classification model.

Second, we study how to design *inherently interpretable* DNNs. In contrast to explaining the models *post hoc*, this approach not only takes the training procedure and the DNN architecture into account, but also modifies them to ensure that the decision process becomes inherently more transparent. In particular, we propose two novel DNN architectures: the CoDA and the B-cos Networks. These architectures are designed such that they can easily and faithfully be summarised by a single linear transformation, and are optimised during training such that these transformations align with the task-relevant input features. As a result, we find that they exhibit a great amount of detail and are able to accurately localise task-relevant features. As such, they lend themselves well to be used as explanations for humans.

Third, we investigate how to leverage explanations to guide models during training, e.g., to suppress reliance on spuriously correlated features or to increase the fidelity of knowledge distillation approaches. In particular, we show that regularising the explanations to align with human annotations or with the explanations of another model can be a powerful and efficient tool to, e.g., improve model robustness under distribution shift or to better leverage limited training data during knowledge distillation.

Finally, in the last part of this thesis, we additionally analyse a popular self-supervised representation learning paradigm: contrastive learning. In particular, we study how a single parameter influences the learning dynamics on imbalanced data and show that it can significantly impact the learnt representations. While not directly linked to model explanations, this work highlights the importance of taking even minor aspects of the optimisation procedure into account when trying to understand and explain DNNs.

ZUSAMMENFASSUNG

In den letzten zehn Jahren haben sich tiefe neuronale Netze (Deep Neural Networks, DNNs) in einer Vielzahl von Anwendungen als äußerst erfolgreich erwiesen. Vor allem in Situationen, die mit hohem Risiko verbunden sind, könnten DNNs daher einen positiven Einfluss auf unsere Gesellschaft haben. So könnten sie aufgrund ihrer herausragenden Fähigkeiten, die oft die des Menschen übertrifft, beispielsweise modernste medizinische Diagnostik für viele leichter zugänglich machen und das allzu oft überlastete medizinische Personal entlasten. Natürlich sind es aber gerade auch solche Situationen, in denen falsche Entscheidungen katastrophale Folgen haben können und möglicherweise Menschenleben gefährden. Insbesondere in diesen Fällen ist es daher unerlässlich, die "Entscheidung" von DNNs erklären zu können.

Diese Dissertation untersucht diese Problematik im Zusammenhang von Bildklassifizierungsmodellen. Zunächst evaluieren wir dafür Methoden, die DNNs "post hoc" erklären und beschreiben sowohl Chancen als auch Unzulänglichkeiten bestehender Ansätze. Zum einen werten wir eine gängige Erklärungsmethode aus und untersuchen damit DNNs, die trainiert wurden zu erkennen, ob ein Gehirnscan von einem gesunden Probanden oder von einem Alzheimerpatienten stammt. Dabei stellen wir fest, dass diese Methode tatsächlich Hirnregionen hervorhebt, die besonders stark von Alzheimer betroffen sind. Leider können wir hierbei jedoch nicht mit Sicherheit sagen, auf welche Aspekte des Eingangssignals sich das Modell stützt. Um Erklärungsmethoden diesbezüglich besser untersuchen zu können, entwickeln wir daher eine neue Auswertungsmethodik, bei der wir explizit kontrollieren, welche Signalmerkmale das Modell mit Sicherheit *nicht* verwendet hat. Damit können wir zeigen, dass einige der beliebtesten Erklärungsmethoden das zu erklärende Modell nachweislich nicht getreu abbilden.

Anschließend widmen wir uns inhärent interpretierbaren DNNs. Im Gegensatz zu post-hoc-Erklärungen werden bei diesem Ansatz das Trainingsverfahren und die Architektur der DNNs nicht nur berücksichtigt, sondern explizit so verändert, dass die Entscheidungsfindung inhärent transparenter wird. In diesem Kontext stellen wir zwei neue, inhärent interpretierbare DNNs vor: die CoDA und die B-cos Networks. Diese Modelle sind dermaßen konstruiert, dass wir die gesamten Modellberechnungen durch eine äquivalente Lineartransformation darstellen können. Während des Trainings wird das Modell weiterhin so optimiert, dass es die resultierenden Transformationen relevanten Strukturen im Eingangssignal angleicht, wodurch diese sich gut als Erklärung der Modellentscheidung für den Menschen eignen.

Darüber hinaus untersuchen wir wie Modellerklärungen genutzt werden können, um Modelle während des Trainings zu leiten, mit dem Ziel die Abhängigkeit der Vorhersage von Kontextmerkmalen (z.B. im Bildhintergrund) zu reduzieren oder Methoden der "knowledge distillation" (KD) zu verbessern. Hierbei werden die Erklärungen dahingehend optimiert, dass sie zusätzlichen Annotationen von Menschen oder den Erklärungen anderer Modelle ähnlich werden. In unseren Experimenten zeigen wir, dass dies ein mächtiges Werkzeug darstellt, um DNNs beispielsweise robuster gegenüber sich verändernder Signalverteilungen zu machen oder um die Modelltreue von KD Methoden zu verbessern.

Schließlich wenden wir uns einem anderen Thema zu und analysieren einen beliebten Ansatz des selbstüberwachten Lernens (dem kontrastiven Lernen) und zeigen, dass ein einzelner Parameter (die "Softmax-Temperatur") die Trainingsdynamik signfikant beeinflussen kann. Auch wenn diese Arbeit nicht in direktem Zusammenhang mit der Erklärung von Modellen steht, unterstreicht sie jedoch, wie wichtig es ist, die einzelnen Faktoren, die den Optimierungsprozess beeinflussen, besser zu verstehen.

ACKNOWLEDGEMENTS

First and foremost, I would like to express my heartfelt gratitude to my advisors Bernt Schiele and Mario Fritz. Thank you for the guidance over the last five years; for providing feedback on all possible levels of detail, from specific word choices, to identifying critical experiments, to asking the right research questions in the first place; for challenging me in our (sometimes intense) discussions and helping me refine my arguments and thoughts; and for your continuous support through the ups and downs of this journey. Looking back at my first days makes me realise how much I learnt along the way and I am incredibly grateful for having been given this opportunity. Thank you Bernt for putting your trust in me, for helping me carve out my path, and for an atmosphere that made it easy to give and receive honest feedback.

I would further like to thank my amazing collaborators: Sukrut, for taking part in a large stretch of my research journey and trusting me to take part in his, for all the discussions and projects we shared, and for being the backbone of our reading group! Anna, for the best from-coffee-break-to-paper experience. Navdeep, for sticking with me through never-ending experiments and for always going the extra mile. Amin, for your close-to contagious perseverance and dedication. Siddhartha, Nhi, Sweta, and Shreyash, for putting your trust in me to take part in your projects, I am very much looking forward to see the fruits of our discussions come to bear. Thanks to all of you, working remotely during the past years has not felt that remote after all.

I also feel blessed to have had the opportunity to work in such an outstanding research environment and would like to thank all my colleagues at the MPI and outside of it for shaping it through their talks, and during discussions, coffee breaks, and retreats. A special shout-out to David, Anna, Sukrut, Jan, Max, Nhi, Mattia, Stephan, Bharat, Eldar, Jovita, Steffen, Fan, Hilde, Christian, Anurag, Sid, Philipp, Yongqin, Jan-Hendrik, Farzaneh, Dengxin, Di, Andrea, Zhi, Li, Yaoyao, Shaoshuai, Gerard, Keyang, Christopher, Margret, Xudong, Tribhu, Xinting, Gul, Nina, Robin, Simone, Saurabh, Rakshith, Wenjia, Ning, Julian, Aymen, Verica, Garvita, Mo, and everyone else. A special thank you to Connie for making everything run smoothly and to the team at the IST for making any technical challenges all but disappear.

Finally, I dedicate this thesis to my friends Martin, Ludwig, and Esther and to my family, this thesis would not have been possible without you. Knowing that you are there for me and I can rely on you is an incredibly powerful source of energy. More than anything, thank you Julia for being at my side throughout this time. No se puede decir en palabras cuanto te agradezco tu apoyo, espero que lo sepas igualmente. ¡Muchas gracias por todo!

CONTENTS

1 INTRODUCTION

Contents

> High-risk AI systems shall be designed and developed in such a way to [...] enable users to interpret the system's output and use it appropriately.
>
> *AI Act Proposal, European Union [Cou21]*

DEEP Learning has come to be the dominant machine learning paradigm, with Deep Neural Networks (DNNs) achieving outstanding performance in tasks ranging from predicting the 3D structure of proteins [JEP+21], to providing diagnoses on par with medical professionals [LJE+20], to autonomously steering vehicles in complex urban environments (cf. [Cal23a, Cal23b]).

This success is often attributed to the fact that DNNs learn predictive features directly from raw data [JGB+21], instead of relying on explicit, hand-crafted features. As a consequence of this, however, their 'decision-making process' is highly opaque and they are thus often described as 'black boxes' [Rud19]. Especially in sensitive contexts such as healthcare or autonomous driving this lack of transparency is a major concern. In fact, in response to this concern, the recently proposed 'Artificial Intelligence (AI) Act' of the European Union would require that "High-risk AI systems shall be designed and developed in such a way to [...] enable users to interpret the system's output and use it appropriately" [Cou21].

This captures well the motivation for this thesis, which, at its core, is concerned with designing DNNs that inherently enable us to better understand their internal computations. Specifically, we propose to rethink the design of DNN architectures and their training procedures to explicitly include the goal of interpretability and show that this allows us to effectively *optimise the DNNs for inherent interpretability*.

In detail, we approach this problem setting from three directions: better understanding the current state of model interpretability (Part I), designing inherently interpretable models from first principles (Part II), and then guiding these models to avoid spuriously correlated features (e.g. background features) and instead rely on object-specific features that are meaningful to humans (Part III).

Additionally, in Part IV of this thesis, we 'zoom out' and, instead of focusing on understanding individual model decisions, aim to better understand the learning dynamics of a popular technique for self-supervised representation learning: contrastive learning. While very different on the surface, this work shares an interesting parallel with the inherently interpretable models of Part II: specifically, we find that a single hyperparameter can play a crucial role in determining the optima the DNNs converge to, thus highlighting the importance of taking the optimisation procedure into account when trying to understand DNNs.

1.1 INTERPRETABILITY IN DEEP LEARNING

While research on understanding machine learning (ML) models pre-dates DNNs [HR19], explainable Artificial Intelligence (xAI) has gained both importance and popularity with the wide-spread adoption of these highly complex models; for an overview of xAI approaches for image classification, see Chapter 2. In short, the goal of xAI is to develop (1) *post-hoc* addenda for existing ML models or (2) novel ML techniques, such that the models' highly complex computations can be faithfully represented in simpler terms in order to provide human understandable explanations for their decisions, cf. [NTP+23]. Before discussing the approaches (1) and (2) in more detail, let us first define the preceding terms more precisely.

Definitions. In the following, we provide short definitions of some of the core concepts used in this thesis.

> **Explanation.** We define an explanation to be a simplified representation of a machine learning model that faithfully describes (an aspect of) that model and is aimed to answer a specific question [GBY+18] in a way that is easily understandable for humans, cf. [NTP+23]. Importantly, the quality of an explanation thus mainly depends on two factors: how easily human-understandable the simplified representation is and how accurately it describes the desired aspect of the model.
>
> Specifically, in this thesis we are primarily interested in answering the question 'Which input features were most relevant for the decision?' for a given DNN-based image classifier.

> **Faithfulness.** To describe how accurately a particular representation reflects a given aspect of a model, we often use the term *faithfulness*; this concept has also been referred to as *completeness* [GBY+18] or *descriptive accuracy* [MSK+19]. E.g., a model is often well approximated locally by a linear function based on its gradient (i.e. the first order Taylor expansion)—the model gradient thus faithfully describes the models' behaviour in a small neighbourhood around a given input.
>
> However, albeit faithful, note that a linear approximation of a model does not necessarily answer the above question regarding feature importance. Instead, it can help answer the question 'To which features is the model most sensitive to if they were to be perturbed slightly?' Moreover, the gradient of DNNs are often highly noisy and are thus not easily understandable for humans.

> **Interpretation.** Within the context of this thesis, an interpretation refers to a simplifying approximation of a complex model such as a DNN. Given the reduced complexity, these representations can then serve as a basis for providing human-understandable explanations (see above). Models which are explicitly designed such that a simplified representation of the model computations exists are thus referred to as *inherently interpretable*. The degree of simplification can, of course, vary. E.g., all model parameters in combination with the model architecture fully describe how the model arrived at its prediction without simplifying the model. In contrast, the highly popular class activation mapping (CAM) method [ZKL+16] yields a drastic simplification of a DNN and approximates the full model by the linear transformation performed by the last layer.

Described in the above framework, a core part of this thesis (Part II) is thus concerned with the design of inherently interpretable DNNs. Specifically, we develop DNNs that can by design be summarised by a single linear transformation for every input. This linear summary thus serves as the model *interpretation*. As this summary is equivalent to the full model computations for every input, it is a *faithful* representation of the models' computations. Finally, we show that these summaries effectively 'reconstruct' task-relevant patterns in the input and thus allow for localising features that were most relevant to the models' decisions. As such, visualisations based on these summaries lend themselves well as *explanations* for humans.

As discussed above, approaches for addressing the black-box nature of DNNs can be divided into two distinct categories: (1) developing *post-hoc explanations* for existing models and (2) designing *inherently interpretable models*. To motivate and contextualise this thesis, we provide a short description of such approaches and how they relate to work performed in this thesis in the following; for details see Chapter 2.

Post-hoc Explanations. In the quest towards better understanding DNNs, a considerable amount of research has been devoted to explaining model decisions in a *post-hoc* fashion (for an overview, see Section 2.1). I.e., the goal of this approach is to develop a general technique for deriving easily understandable explanations from any pre-trained DNN. This has an obvious appeal: it decouples the model performance on the main task of interest (e.g. classification accuracy) from its interpretability, thus simplifying model development. Moreover, if such a method were to be found, the *black box problem* of DNNs would effectively be solved and past, present, and potentially also future applications of DNNs might at once become more transparent. The research on post-hoc explanations has yielded many promising approaches, some of which have been shown to provide explanations (e.g. in the form of feature importance heatmaps) that correlate with meaningful signals in the data, e.g. in chemistry [SGTM19], pharmacology [PKR+19], or histopathological images [BBH+21], see also [SML+21].

Similarly, in Chapter 3 we analyse DNNs trained for detecting Alzheimer's disease (AD) from magnetic resonance imaging (MRI) data. In this context, we show that a popular method for interpreting DNN models, namely layer-wise relevance propagation (LRP) [BBM+15], yields explanations that emphasise brain areas known to be particularly affected by AD, thus highlighting a major promise of xAI research. In particular, if we can reliably explain DNNs, we might eventually learn from these models and, e.g., identify new biomarkers that could allow us to detect and treat medical conditions at an earlier stage.

Unfortunately, it is generally difficult to evaluate whether a given model interpretation (e.g. via LRP) indeed yields reliable explanations, as we do not know the *true reasons* that lead a DNN to a specific classification—after all, this is the problem that post-hoc methods aim to solve in the first place. Consequently, evaluating the correctness of explanations and comparing them fairly has proven challenging.

In Chapter 4, we make a step towards addressing this issue. In particular, we carefully construct an evaluation setting in which certain input features are known to *not contribute* to the model decision and develop a framework that allows for comparing different explanation methods in a systematic and fair manner. Interestingly, we find that some popular post-hoc explanation methods can attribute importance to features that—by construction—do not influence the DNNs, thus putting their faithfulness into question.

This highlights a common concern regarding post-hoc explanations: can (and should) we trust a fixed approximation method of black box models to reliably provide faithful and understandable explanations of the underlying decision process, cf. [Rud19]? In fact, can the simplifying assumptions of the model interpretations be expected to hold *independent* of the training paradigm? And, even if they do, can we expect the resulting explanations to be easily understandable for humans?

In fact, at least two commonly used explanation methods (gradients [TSE+19] and self-attention [CTM+21]) have been shown to yield drastically different results depending on the training paradigm. In light of this, we argue that the optimisation procedure of DNNs should be taken into account when explaining DNNs. Even more, we propose to design the optimisation procedure so that the models become inherently interpretable and yield explanations that lend themselves well to be easily understandable for humans.

Inherently Interpretable Models. One of the core challenge in designing inherently interpretable models is to ensure that they not only provide understandable and faithful explanations, but also achieve performance competitive with non-interpretable models (see also the desiderata put forward in [MSK+19]). While it is often assumed that an inherent trade-off between interpretability and performance exists, others

argue that interpretability might even help to improve models [Rud19, RCC+22].

In Part II of this thesis, we make a step towards designing performant, yet inherently interpretable models and show that performance and interpretability might indeed not necessarily be at odds. For this, we propose to rethink the model design and optimisation procedure with the explicit goal of increasing model interpretability. In particular, we design models that inherently allow to be faithfully summarised by a linear transformation, which is optimised during training to align with relevant input features and thus lends itself well to be easily understandable for humans.

While this has been attempted before [AJ18], to the best of our knowledge, our models are the first to perform well with respect to all three of the following desiderata: they (1) are performant and (2) yield faithful model explanations that (3) lend themselves well to being understandable to humans. E.g., piece-wise linear models (e.g. ReLU-based [NH10] DNNs) are performant and faithfully summarised by a linear transformation for every input, but the resulting explanations are not easily understandable. The Self-Explaining Neural Networks by [AJ18], on the other hand, are optimised to yield stable and thus more easily understandable explanations, but sacrifice predictive performance. The B-cos Networks we propose in Chapter 6, in contrast, achieve high classification performance *and* provide highly detailed, human-understandable explanations that faithfully summarise the model computations for every input.

1.2 Outline and Author's Contributions

This thesis is divided into 5 parts and a total of 10 chapters. In the following, we provide a short overview over the content of each of the remaining 9 chapters. Further, we highlight the publications in which the respective content has previously appeared; in this context, '*' denotes equal contributions of the leading authors. We would like to further emphasise that the senior advisors Bernt Schiele, Mario Fritz, Hilde Kuehne, Christian Rupprecht, and Kerstin Ritter provided invaluable feedback throughout all the development stages of the corresponding projects, and we thank them for their continuous support.

Chapter 2 — Related Work. In this chapter, we provide a detailed overview over the relevant literature to contextualise this thesis appropriately. To paint the full picture, this includes *prior* work by which this thesis is inspired and on which it is built, *concurrent* work that appeared alongside the approaches we develop and discuss in this thesis, as well as *follow-up* work that has appeared since and which allows us to identify important open questions and promising recent developments, see also Chapter 10.

1.2.1 Part I — The Promises and Pitfalls of Attribution Methods

In the first part of this thesis we focus on evaluating existing explanation methods. By doing so, we highlight both the promises (Chapter 3) and open challenges (Chapter 4) of explanation methods.

Chapter 3 — The Promises. As a starting point for this thesis, in this chapter we present a case study on understanding the decisions of Deep Neural Network (DNN) classifiers in the medical domain. Specifically, we use the layer-wise relevance propagation (LRP) technique [BBM+15] to analyse DNNs that were trained to detect signs of Alzheimer's disease (AD) in magnetic resonance imaging (MRI) data. In this context, we showcase and discuss some of the potential benefits of explaining the DNNs' decisions. The corresponding publication [BEWR19] has previously appeared in Frontiers in Aging Neuroscience.

As one of the lead authors of [BEWR19], Moritz Böhle developed the majority of the code, the experiments, and the quantitative and qualitative evaluation procedures. The general study design and the literature review were done jointly with Fabian Eitel. The majority of the writing for the first submission was done

by Moritz Böhle, whereas the majority of the revisions of the manuscript after the first (and last) round of reviews were done by Fabian Eitel. Fabian Eitel was further responsible for revising the code for publication, as well as the design, implementation, and evaluation of relevant ablation studies.

Chapter 4 — The Pitfalls. Given the importance of making the decision processes of DNNs more transparent, a wide range of different approaches for doing so have been developed; for a detailed discussion, see Chapter 2. In order to better understand the differences between methods from a highly popular subgroup—so-called importance attribution methods—in this chapter we perform a systematic evaluation across a wide range of approaches. By doing so, we are able to distill and highlight various important shortcomings of existing approaches, such as a lack of model faithfulness. Moreover, we show that apparent performance differences between some of the most widely used methods can be attributed to the fact that the comparisons are often skewed. To address this, we provide a framework to more fairly compare attribution methods, both qualitatively and quantitatively. This work has previously appeared at CVPR ([RBS22]) and an extension has been accepted for publication at IEEE TPAMI ([RBS24]).

For [RBS22, RBS24], as second author, Moritz Böhle contributed significantly to the study design, the evaluation procedure, and the general development of the project. All the experiments and their implementation, their quantitative and qualitative evaluation, as well as the development and publishing of the code were done by Sukrut Rao. Moritz Böhle and Sukrut Rao contributed in equal parts to the writing.

Publications

[BEWR19] **Moritz Böhle***, Fabian Eitel*, Martin Weygandt, Kerstin Ritter. Layer-wise Relevance Propagation for Explaining Deep Neural Network Decisions in MRI-based Alzheimer's Disease Classification. In *Frontiers in Aging Neuroscience, 11*, 2019.

[RBS22] Sukrut Rao, **Moritz Böhle**, Bernt Schiele. Towards Better Understanding Attribution Methods. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[RBS24] Sukrut Rao, **Moritz Böhle**, Bernt Schiele. Better Understanding Differences in Attribution Methods via Systematic Evaluations. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2024.

1.2.2 Part II — Designing Inherently Interpretable Deep Neural Networks

In Part I, we focused on analysing existing attribution methods that were developed to understand the decision-making process in DNNs *post hoc*, i.e. independently of the training or the design of the DNNs.

In contrast, in Part II we place a particular focus on developing *inherently interpretable* DNNs. Specifically, we show that it is possible to include the goal of interpretability into model design and training, resulting in DNNs that are simultaneously optimised for classification performance and interpretability.

Chapter 5 — CoDA Networks. In this chapter, we introduce the notions of *dynamic linearity* and *alignment pressure* as tools for building inherently interpretable DNNs, and based on this, design the Convolutional Dynamic Alignment (CoDA) Networks. We show that these models indeed exhibit a high degree of interpretability and can constitute performant classifiers. The content of this chapter has previously been presented at CVPR (oral) [BFS21] and has been published at IEEE TPAMI [BFS22a].

As the lead author, Moritz Böhle designed the proposed models, conducted all the experiments, and was the main writer for [BFS21, BFS22a].

Chapter 6 — B-cos Networks. While the CoDA Networks showed promising gains in interpretability and competitive performance, they do not easily scale to complex, large-scale classification problems. In this chapter, we present a drastically simpler model that nonetheless adheres to the design principles of the CoDA Networks, namely the dynamic linearity and the alignment pressure. Importantly, we show that the core components of the resulting B-cos Networks are compatible with a wide range of conventional DNNs and not only significantly increase their interpretability, but also maintain their classification performance on challenging datasets. The content of this chapter has previously appeared in CVPR, see [BFS22b], and has recently been accepted for publication at IEEE TPAMI [BSFS24].

As the lead author, Moritz Böhle designed the proposed models, developed the code base, conducted all the experiments, and was the main writer for [BFS22b]. For the extension to [BSFS24], Moritz Böhle was the main writer and responsible for the experimental and model design, as well as the qualitative evaluations. Navdeeppal Singh implemented the majority of the experiments, and was the main contributor for revising the code from [BFS22b], as well as for extending and publishing it.

Publications

[BFS21] **Moritz Böhle**, Mario Fritz, Bernt Schiele. Convolutional Dynamic Alignment Networks for Interpretable Classifications. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[BFS22a] **Moritz Böhle**, Mario Fritz, Bernt Schiele. Optimising for Interpretability: Convolutional Dynamic Alignment Networks. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2022.

[BFS22b] **Moritz Böhle**, Mario Fritz, Bernt Schiele. B-cos Networks: Alignment is All We Need for Interpretability. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[BSFS24] **Moritz Böhle**, Navdeeppal Singh, Mario Fritz, Bernt Schiele. B-cos Alignment for Inherently Interpretable CNNs and Vision Transformers. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2024.

1.2.3 Part III — From Understanding Models to Guiding Models

In the parts, we consider explanation methods as a tool that allows to provide additional information about the models to humans. In this part, we explore the opposite direction, i.e. how explanations can be used to provide additional information *to the models*.

Chapter 7 — Guiding Models. In Chapter 7, we do so by providing additional human annotations (object bounding boxes) to the models during training and regularise the model explanations to specifically highlight the corresponding regions in the input images. In particular, we perform a detailed study across a wide range of design choices (attribution methods, models, loss functions, and other hyperparameters) on two large-scale datasets, to better understand how this can be done efficiently and effectively. The corresponding publication [RBPAS23] has previously appeared at ICCV.

Moritz Böhle and Sukrut Rao contributed the majority and in equal parts to the writing and the study design of [RBPAS23]. The implementation of the code for training and quantitatively evaluating the models was done by Sukrut Rao, the data analysis was done jointly by Sukrut Rao and Moritz Böhle, and the data and explanation visualisations were created primarily by Moritz Böhle. The experiments on the Waterbirds dataset were conducted by Amin Parchami-Araghi.

Chapter 8 — Explanation-enhanced Knowledge Distillation. In addition to providing information to the models via human annotations, we investigate the benefits of providing additional information to DNNs via the explanations of another model. In particular, we show that regularising the explanations of 'student' models to be similar to those of the 'teacher' models can help distill knowledge between models more efficiently and improves distillation fidelity; i.e., student models behave more similarly to their teachers if explanation distillation is applied, maintaining important properties such as robustness to distribution shifts. The corresponding work is currently under review.

Moritz Böhle, Amin Parchami-Araghi, and Sukrut Rao contributed in equal parts to the study design and the writing of [PABRS23]. All experiments were conducted and implemented by Amin Parchami-Araghi.

Publications

[RBPAS23] Sukrut Rao*, **Moritz Böhle***, Amin Parchami-Araghi, and Bernt Schiele. Studying How to Efficiently and Effectively Guide Models with Explanations. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[PABRS23] Amin Parchami-Araghi*, **Moritz Böhle***, Sukrut Rao*, and Bernt Schiele. Good Teachers Explain: Explanation-Enhanced Knowledge Distillation. Under review.

1.2.4 Part IV — Contrastive Learning on Long-Tail Data

Chapter 9 — Improving Contrastive Learning on Long-Tail Data. In the last chapter on work performed for this thesis, we switch gears and, instead of analysing individual model decisions, try to better understand the mechanics of a commonly used framework for self-supervised representation learning. In particular, we study the effect of a single hyperparameter of contrastive learning methods: the softmax temperature. We show that this parameter critically influences the learning dynamics and that periodically changing it throughout the training process yields significant benefits for the learnt representations on long-tail datasets. This work has previously appeared at ICLR, see [KBS+23].

Moritz Böhle and Anna Kukleva contributed in equal parts to the design of the temperature scheduling approach, the analytical framework, as well as the writing of [KBS+23]. Initial results indicating a significant effect of the temperature parameter were obtained by Anna Kukleva. The implementation of the experiments and the publication of the code for this project were fully handled by Anna Kukleva.

Publications

[KBS+23] Anna Kukleva*, **Moritz Böhle***, Bernt Schiele, Hilde Kuehne, Christian Rupprecht. Temperature Schedules for Self-Supervised Contrastive Methods on Long-Tail Data. In *International Conference on Learning Representations (ICLR)*, 2023.

1.2.5 Conclusion and Outlook

Chapter 10 — Conclusion and Outlook. In this chapter we summarise the most important findings and contributions from this thesis and discuss them against the backdrop of recent developments in the field. In this context, we highlight important challenges that remain and how these could be addressed.

2 Related Work

Contents

In the following, we highlight prior works that share important parallels with and served as inspiration for the methods we develop in this thesis. In particular, to put our work on evaluating post-hoc explanation methods (Part I) and on designing inherently interpretable Deep Neural Networks (DNNs) (Part II) into context, we will first review related research on explainable AI (xAI) for image classification; for a broader overview of xAI research, see e.g. [BPBPQE23, AAES+23]. In particular, we discuss related work on understanding DNN-based image classifiers via post-hoc explanations in Section 2.1 and on evaluating the quality of such explanations in Section 2.2. Then, in Section 2.3, we describe recent efforts on designing inherently interpretable DNNs, which are the core focus of Part II. Moreover, to contextualise our work on guiding models by regularising their explanations (Part III), we discuss relevant related work on model guidance and knowledge distillation approaches in Section 2.4. Finally, in Section 2.5, we highlight similarities of prior work to our 'temperature scheduling' approach, which we develop to improve self-supervised representation learning on long-tailed data (Part IV).

2.1 Post-hoc Explanations of Deep Neural Networks

Commonly, xAI research for DNN-based classification models aims at better understanding the individual predictions of *existing* classifiers. As the design and optimisation of the classification models are thus decoupled from the question of how to explain them, the resulting explanations are often referred to as *post-hoc*. In the following, we describe the most prominent examples of this in detail and how they relate to the methods we present in this thesis. In particular, we first introduce importance attribution methods in Section 2.1.1. Thereafter, in Section 2.1.2, we discuss approaches that aim to shed light on intermediate model representations. In this context, we also highlight recent works on providing global explanations, i.e., methods that describe model behaviour *across images* instead of providing *per-image* explanations.

2.1.1 Explaining Model Decisions via Importance Attributions

Post-hoc importance attribution methods can be divided into either *white-box* or *black-box*, which differ in the level of access to the model they require. Specifically, white-box attribution methods assume full access to the internal model computations, such as intermediate activations, the models' gradients, and the model architectures in general. In contrast, black-box attribution methods only require access to the model outputs for differently perturbed inputs, as we discuss in the following. We then highlight a few

selected approaches that exhibit interesting parallels to the methods developed in this thesis.

White-box Approaches are typically *backpropagation-based* and combine the model inputs and the (modified) gradients of the models' outputs with respect to those inputs in different ways and to varying degrees [SVZ14, SDBR15, ZF14, SGK17, STY17, SF19, ZBL+18, BBM+15, MBL+19, ZKL+16, SCD+17, CSHB18, JZH+21, DR20, WWD+20]; note that the methods are often applied to just a part of the model (e.g. the last layer as in [ZKL+16, SCD+17]) and the 'input' in that case is thus given by the input to the first layer of the respective submodel. Conceptually, these approaches are thus highly related and primarily differ in how specific rules are chosen for modifying the gradients or weighting the inputs. E.g., Sensitivity [SVZ14] uses the gradient itself, whereas [ZF14, SDBR15, BBM+15, MBL+19] introduce various rules by which gradients are altered in the backwards pass through the model. Going further, Layer-wise Relevance Propagation (LRP) [BBM+15, MBL+19] explicitly turns the rules into a design choice and thus constitutes a general framework that encompasses many other approaches. Further, some methods [SGK17, STY17] introduce additional reference values on which the rules depend [SGK17] or to improve the gradient-based importance estimates [STY17], or even try to learn the optimal backpropagation strategy from data [KSA+18]. Finally, note that the group of so-called 'class activation map' (CAM) approaches also typically relies on modified gradients to weight the inputs to a specific layer of the model [ZKL+16, SCD+17, CSHB18, JZH+21] and are thus conceptually highly related to computing 'Input$\times$Gradient' (cf. [SGK17]); that said, some CAM approaches weight the layer inputs by means of a reference value [WWD+20] or by perturbing the inputs to the layer [DR20].

Black-box Approaches are typically *perturbation-based*, i.e. they estimate the importance of input features by evaluating the model on many perturbed versions of an input [RSG16, PDS18, ZF14, FV17, DG17]. This is done by iteratively replacing different parts of the input with a fixed value (e.g. an average value) [RSG16, PDS18, ZF14, LL17] or optimising a mask over the input image that maximises / minimises class confidence [FV17, DG17]. While such black-box approaches are highly versatile, they are not based on a *mechanistic understanding* of how the underlying model computes its output. Instead they can be seen to answer the question: "If perturbed, which input features most strongly affect the models' output?" In contrast, in Chapters 5 and 6 we explicitly aim to design models for which we better understand the underlying mechanism, which in turn allows us to derive explanations directly from the model itself.

In Chapter 4, we compare both white- and black-box approaches in a systematic manner to better understand differences between the methods and highlight failure modes. Specifically, for this we group the approaches into being primarily gradient-based, activation-based (i.e. CAM methods), or perturbation-based, and analyse their behaviour when applied to different layers of the models under investigation.

Relation to CoDA and B-cos Explanations. In Chapters 5 and 6, we introduce novel DNN architectures (CoDA and B-cos Networks respectively), which compute their outputs in an easily interpretable manner. For this, the models are designed to be *dynamic linear*: i.e., the output is given by an input-dependent linear transformation of the input ($\mathbf{y} = \mathbf{W}(\mathbf{x})\mathbf{x}$). The resulting linear transformations are hence *white-box* explanations, as they are explicitly part of the model architecture and depend on the internal computations. While they thus share similarities with the white-box approaches above, there are also crucial differences.

First, we note that the linear decomposition of the output into linear contributions from individual input dimensions is equivalent to the Input$\times$Gradient method (cf. [SGK17]) for piece-wise linear models. Further, the explanations for CoDA and B-cos Networks are obtained via backpropagation: for this, we modify the actual model gradients to reflect the dynamic linear transformations of each individual layer. In fact, the rules by which the gradients are modified can be considered to be the appropriate LRP rules for

the respective layers, as the total relevance is naturally conserved in the CoDA and B-cos explanations[1]. Importantly, however, in contrast to the model-agnostic, post-hoc explanations discussed above, the CoDA and the B-cos networks are explicitly designed and optimised to be explainable under this particular form of backpropagation and the resulting explanations are thus not *post-hoc*, but rather *model-inherent*.

2.1.2 Extracting Learnt Concepts and 'Global' Explanations

In the previous section, we discussed methods that aim to highlight which input features were most influential for the models' *output* layers. In contrast, in this section we highlight approaches that aim to improve our understanding of which 'concepts' *intermediate* layers in DNNs represent and how this can help to describe global model behaviour, i.e., across images, instead of providing per-image explanations.

Neuron Visualisations. [HOWT06] show that first layer neurons of DNNs trained on natural images can learn Gabor-like filters [Gab46, Gra78] by visualising their weight matrices. Extending such visualisations to higher-level neurons, [EBCV09] introduced the idea of activation maximisation, which optimises input images to maximally activate a chosen neuron. By regularising these images to conform with certain priors (for a discussion, see [OMS17]), these optimised images can exhibit semantically understandable patterns, such as faces or car wheels. This approach is still commonly used, cf. [OCS+20, GCV+21], but has recently been shown to result in images that are processed very differently by the DNNs than natural images [GZB+24] (i.e. from the original data distribution), which puts their reliability for explaining model behaviour into question. Similarly, in Chapter 6, we visualise intermediate neuron explanations as natural images. However, instead of optimising the input, we directly visualise the dynamically computed linear mapping of the input to the chosen neuron. As such, our visualisations do not rely on hand-crafted priors and are not based on out-of-distribution samples, such as the optimised images in activation maximisation.

Extracting Learnt Concepts. Various approaches for mapping intermediate features of a DNN to human-interpretable concepts have been proposed. E.g., [BZK+17, ZKL+15] use correlations of intermediate activations with annotated semantic concepts to show that some neurons consistently 'fire' for human-interpretable concepts. Instead of only relying on correlations, [KWG+18] use an annotated concept dataset and train linear classifiers to map the intermediate representations of a DNN to those human-understandable concepts. Similarly, [CBR20] 'whiten' the intermediate features and rotate the representations such that individual axes align with pre-defined human-interpretable concepts. Additionally, a recent line of work explores how to extract the concepts on which the model relies in an unsupervised fashion and without relying on human annotations [GWZK19, ZMM+21, FPB+23, GNO+23], by means of clustering features [GWZK19] or by applying matrix factorisation [ZMM+21, FPB+23, GNO+23, FBB+23].

We note that approaches for concept extraction are complementary to our work on inherently interpretable models (Chapters 5 and 6) and leave a detailed analysis of the intermediate features of the CoDA (Chapter 5) and B-cos networks (Chapter 6) to future work. That said, in Chapter 6 we provide preliminary evidence that the intermediate features can be mapped to human-interpretable concepts (such as wheels, bird beaks, etc.) by qualitatively analysing the model-inherent explanations for those features.

From Local to Global Explanations. Finally, we would like to highlight a particularly interesting aspect of combining concept-extraction techniques and importance attribution methods as discussed in the previous section. Specifically, recent work explores the reliance of the final network prediction on such extracted intermediate concepts [KWG+18, FPB+23, FBB+23, ADE+23], by measuring the output

[1] I.e., by design, the linear contributions obtained for CoDA and B-cos models sum to the output logit, thus conserving the total amount of 'relevance' (logit score). See also the conservation principle for layer-wise relevance propagation in [BBM+15].

change under 'concept removal' [FPB+23, FBB+23], or by applying backpropagation-based importance attribution techniques [KWG+18, ADE+23, FBB+23]. By aggregating the concept importance estimates over many images, it becomes possible to make global statements about a models' reliance on the intermediate concepts. Specifically, if intermediate concepts are human-understandable and can be labelled (e.g., 'striped', 'wheel', 'face', or 'watermark'), one can assess the strategies of the models for solving a task across images and potentially detect unwanted model behaviour more easily (e.g., "The detection of class A often relies on spuriously correlated watermarks in the training dataset"). While this is thus a highly promising research direction, we also note that the reliability of such statements crucially hinges on good estimates for the importance of individual concepts. By improving the importance estimates or analysing the reliance on concepts for inherently interpretable models (as in Chapters 5 and 6), the global explanations might better reflect the true model behaviour and become more easily understandable.

2.2 Evaluating Explanations of Deep Neural Networks

As discussed in the previous section, many different approaches for better understanding the predictions of DNNs have been proposed in recent years. However, given that the true reasons for a given DNN decision are typically not known, it is often challenging to assess the quality of the resulting explanations. To overcome this, the development of explanations has been accompanied by the development of an array of evaluation techniques; for an excellent recent review, see [NTP+23]. Generally, these techniques test for desirable properties of explanations. In the following, we discuss *human-centric evaluations* [KMR+22, CFCS22, SSP+22, SH20, HB20], *sanity checks* [AGM+18], *localisation-* [ZBL+18, SSTL20, CLY+15, FV17, BFS21, YK19], and *perturbation-based metrics* [SBM+16, SF19, CSHB18, HSMR23].

Human-centric Evaluations. A core motivation for developing explanations for DNNs is to help humans better understand how a given model arrived at its prediction. To assess whether explanations live up to this promise, various evaluation protocols involving human subjects have been proposed. In this context, a common aspect of the explanations that is tested for is that of 'simulatability' [HB20]. Specifically, simulatability assessments evaluate whether explanations help humans to predict model behaviour. While some of the tested explanations do help users predict the model behaviour in simple settings [RSG16, CFCS22], similar benefits have not been observed on more complex tasks [KMR+22, CFCS22, SH20].

Moreover, [KMR+22] find that accompanying model predictions by explanations significantly increases the users' trust in the underlying models. In fact, users are willing make significant sacrifices with respect to model performance if a lower-performing model additionally provides explanations for its predictions, especially in high-risk settings. While this shows that explanation methods could play a crucial role in increasing the adoption of machine learning models, it also highlights the dangers of explanation methods that are not faithful to the underlying model. Specifically, such explanations could engender a false sense of trust in the DNNs' predictions, without actually providing reliable information about their 'reasons'.

Sanity Checks. To mitigate the issue of potentially relying on explanations that do not faithfully describe the DNN in question, [AGM+18] develop two sanity checks that a model-faithful explanation should necessarily fulfil. In particular, the authors evaluate various attribution methods for classification models. Given that these methods are supposed to explain the decision of a *given model* trained for a *specific task*, the authors test whether the attribution methods are in fact dependent on the model and the task. Interestingly, they find that some explanation methods are surprisingly stable with respect to randomising the model parameters and the labels used during training, putting the faithfulness of those methods into question. Similarly, in Chapter 4, we develop an evaluation setting for attribution methods that also

constitutes a sanity check for the methods' faithfulness. Instead of basing it on the methods' model-specificity as in [AGM+18], we carefully control the flow of information through the model, which allows us to make definitive statements about which regions in the input cannot possibly have contributed to the output—any method that still assigns importance to those regions is thus provably not faithful.

Localisation-based Metrics. Instead of evaluating the utility of explanation methods via human studies and as a proxy for evaluating the faithfulness of the explanations, various localisation metrics have been developed [ZBL+18, SSTL20, CLY+15, FV17]. A common assumption for these metrics is that the model will rely on class-specific 'on-object' features; i.e., the correctness of explanations is evaluated by assessing whether an explanation method (primarily) assigns high importance values to pixels contained within human-annotated masks for the target objects (pixel-level segmentations or bounding boxes). This assumption might seem reasonable and will often be valid for a model that performs well on a complex classification task. However, it also arguably defeats the purpose of trying to understand what the *model* does, as it lifts the *human expectations* to be the gold reference (cf. [NTP+23], Sec. 6.11). It thus carries the risk of confirmation bias: such metrics favour explanations that fulfil our expectations and penalises explanations that faithfully highlight that the model relies on spuriously correlated background features.

To overcome this, in Chapters 4 and 5 we introduce a novel evaluation paradigm based on synthetically created *image grids*; similar paradigms are also explored in [SJN21, ADPGG21]. In these grids, each cell belongs to a different class and, when passed individually through the network, it is correctly and confidently classified by the model. As such, we know that each grid cell contains features that allow the model to correctly classify that image. For each of the present classes, we then measure the fraction of importance attribution values that a given explanation method assigns to the grid cell. Importantly, this approach thus does not distinguish between on-object and background features. In the framework of [NTP+23], our proposed paradigm thus evaluates the contrastivity (i.e. target specificity) of the explanations and can be considered a variation of a 'controlled synthetic data' experiment.

However, this approach can also not *guarantee* that the model only uses information from within the desired grid cell, as the class logits can generally be influenced by features from all grid cells. Therefore, in Chapter 4, we additionally propose a highly controlled setting in which we ensure that the logit for each of the present classes is only connected to its respective cell in the grid. As a result, we can guarantee that grid cells are classified independently, allowing us to identify definitive failures of attribution methods.

Perturbation-based Metrics typically test whether the importance attributions as assigned by a specific explanation method are consistent for similar inputs [AJ18, DAA+19] or predictive of the model behaviour when applying perturbations to the input (cf. 'stability' and 'deletion/addition' protocols in [NTP+23]). E.g., in the so-called 'deletion protocol' [SBM+16, SF19, FV17, PDS18], it is assumed that if an input feature (e.g. an image pixel) is assigned a high importance value by an explanation method, the models' confidence in the prediction should drop most strongly when replacing that feature by a baseline value; vice versa for the 'insertion protocol' [PDS18], i.e. it is assumed that the model's prediction should be recovered most quickly when inserting the most important features to a chosen baseline image.

While perturbation-based metrics thus test for desirable properties of explanations in terms of human expectations ('similar inputs should have similar explanations' and 'removing supposedly important features should reduce the model confidence most') they do not necessarily test the explanations' faithfulness. E.g., if a DNN is highly sensitive to image perturbations, it is unclear whether similar explanations can be expected for (albeit slightly) different inputs. Moreover, such interventions inherently depend on the chosen baseline values and can introduce artefacts that are out of the training distribution (OOD). While [HEKK19] mitigate this by retraining a model on data in which the purportedly most important features are removed and then evaluating how much this impacts performance, this approach can be very costly for

large models on complex datasets. Moreover, rather than testing whether a given explanation is faithfully reflecting a particular model, this approach tests whether the combination of model and explanation allow for identifying features that most drastically impact the predictive performance of a new model.

That said, in Chapter 5, we find the model-inherent explanations of the CoDA Networks to be highly predictive of model behaviour under perturbations. Similarly, in a recent work on a synthetic dataset [HSMR23]—which thus ensures that the interventions do not yield OOD data—the authors find that the explanations of the B-cos Networks we introduce in Chapter 6 also exhibit desirable properties under a part-based feature removal paradigm and compare favourably to other explanation methods.

2.3 Designing Inherently Interpretable Deep Neural Networks

In contrast to techniques that aim to explain models *post-hoc*, some recent work has focused on designing new types of network architectures, which are *inherently* more interpretable. To achieve this, inspiration is often taken from simpler models such as decision trees [WDH+21, AMSA21, HCLR19] or linear classifiers. As we pursue the latter, we discuss the respective works in more detail in the following.

In general, approaches that leverage linearity to increase the interpretability of DNNs typically use at least one of the following two ingredients: (I) introducing interpretable *bottlenecks*, on which linear classifiers are trained to produce the output, or (II) using what we will henceforth call *dynamic linear classifiers*.

(I) Interpretable Bottlenecks. Among the most popular approaches that leverage the idea of interpretable bottlenecks are prototype-based networks as introduced in [LLCR18, CLT+19]. The key idea for this approach is to first train a feature extractor to detect 'prototypical parts' in a given input image, which are then classified by a single linear layer. As a result, the exact contributions of each detected prototype to the final prediction are known. By using a decoder [LLCR18] or constraining the prototypes to be given by a specific image patch from the training set [CLT+19], one can further visualise the prototypes as natural images to which any new image is compared. This approach has been extended multiple times, for example by using deformable prototypes [DBC22], defining prototypes as basis vectors [WLWJ21], exchanging the linear classifier by a decision tree [NVBS21], or adapting it to segmentation tasks [SRS+23]. We further note that these prototype-based approaches are conceptually highly related to *concept bottleneck models* [KNT+20, YWZ23, LFS21] (if the bottleneck is applied to the penultimate layer), which are becoming increasingly popular, especially in combination with vision language models [MV23, PIM23, ODNW23].

While the aforementioned approaches often allow to make exact statements about the contributions of the prototypes to the prediction, the prototypes themselves are typically not easily interpretable [HFRK21, KMR+22, NVBS21]; the DNN responsible for extracting the prototypes remains unexplained and the patch similarity in the models' feature space often does not coincide with human notions of patch similarity [KMR+22]. In contrast, in the models we introduce in Chapters 5 and 6 the full computation from input to output is accounted for in the explanations. That said, note that our models could potentially complement bottleneck-based approaches, as these might benefit from using more interpretable feature extractors. This constitutes a promising research direction, which we intend to explore in future work.

Lastly, we note that the BagNets introduced in [BB19] also derive their interpretability from a 'bottleneck'. Specifically, in [BB19] the authors severely restrict the receptive field of the models and linearly classify the representations of relatively small patches independently. As a result, the exact linear contributions of each patch are known and can be attributed to a small area in the input image. Similarly, in Chapters 5 and 6 we also modify the model architectures to increase their interpretability. Instead of restricting the receptive field, however, we propose novel types of network layers which change the optima of the models in a targeted manner and allow us to assign linear contributions to individual pixels.

(II) Dynamic Linear Models. Most related to the models we propose in Chapters 5 and 6 are approaches that derive their interpretability from computing the model output $\mathbf{y}(\mathbf{x})$ in a *dynamic linear fashion*. I.e., the output $\mathbf{y}(\mathbf{x})$ is computed as an input-dependent linear transformation of the input $\mathbf{x}$: $\mathbf{y}(\mathbf{x}) = \mathbf{W}(\mathbf{x})\mathbf{x}$, with the linear transformation $\mathbf{W}(\mathbf{x})$ serving as the model interpretation (see also Section 1.1). The most prominent prior work example of this are the self-explaining neural networks (SENNs) [AJ18], which are motivated very similarly to our work. Specifically, the SENNs also derive their explanations from a linear decomposition of the output into contributions from the input (features). In contrast to our work, [AJ18] use a DNN—which is itself not interpretable—to predict the matrix $\mathbf{W}(\mathbf{x})$, whereas this matrix is constructed layer by layer in our models introduced in Chapters 5 and 6; importantly, each of these layers is itself designed to be easily interpretable.

Moreover, *dynamic linearity*, i.e., the property that the output is computed via some form of an input-dependent linear mapping, is additionally shared by all piece-wise linear networks (e.g., ReLU-based [NH10] networks). In fact, the contribution maps of CoDA and B-cos models are conceptually similar to 'Input $\times$ Gradient' (IxG) [SGK17] for piece-wise linear models, which also yields a linear decomposition in form of a contribution map. In contrast to the piece-wise linear functions, we combine this *dynamic linearity* with a structural bias towards an alignment between the contribution maps and discriminative patterns in the input. This results in explanations of much higher quality, whereas IxG on piece-wise linear models has been found to yield unsatisfactory explanations of model behaviour [AGM+18, KSA+18].

Finally, we note that attention layers as used in transformers [VSP+17, DBK+21] are also dynamic linear and the resulting attention matrices $\mathbf{W}(\mathbf{x})$ are often used to better understand attention-based models. However, in contrast to our models, which only use dynamic linear layers and thus become dynamic linear as a whole, transformers employ attention layers and multi-layer perceptrons (MLPs) in an alternating fashion. Conventional transformers can thus not be summarised faithfully via a single dynamic linear transformation. Since the attention matrices alone do not capture the computations of the full models [BF20, CGW21] additional mechanisms are required to explain the combined computations of attention and MLP layers, such as [CGW21]. In contrast, in Chapter 6 we show that it is possible to construct fully dynamic linear transformers by combining attention layers with B-cos MLPs.

Additional Similarities to CoDA and B-cos Networks. In Chapters 5 and 6, we design novel network architectures with the explicit goal of making them inherently interpretable. For this, we replace the ubiquitously used linear transformations in Deep Neural Networks (DNNs) by Convolutional Dynamic Alignment (CoDA, Chapter 5) and B-cos transformations (Chapter 6) respectively. Given the central role that the CoDA and B-cos models play in this thesis, in the following we highlight and discuss interesting parallels to other work in more detail.

First, in our models, the convolutional kernels are dependent on the specific patch that they are applied to; i.e., CoDA and B-cos layers effectively apply different filters at every position in the input. As such, they can be regarded as an instance of dynamic local filtering layers as introduced in [JDBTG16]. Importantly, we design these dynamic local filters such that they are dynamic linear (see above) and structurally biased towards yielding dynamic linear weights that exhibit a high cosine similarity with relevant input patterns.

Second, the CoDA networks are related to capsule networks as proposed in [SFH17]. In fact, the dynamic weights predicted by the CoDA networks are equivalent to the activations of capsules after a single iteration of the dynamic routing mechanism (for details on dynamic routing, see [SFH17]). However, while in [SFH17] the activations of the capsules directly serve as input to the next layer, in CoDA networks the corresponding vectors are used as convolutional filters; for a detailed discussion, see Appendix B.

Lastly, we note that the B-cos transformations constitute an approximation to radial basis function (RBF) layers with isotropic Gaussian kernels on unit norm hyperspheres. As such, B-cos networks as a whole

can be regarded as an approximation of deep RBF networks; for a discussion, see Appendix C. While it has been argued that RBF networks could exhibit desirable properties [GSS15], integrating RBF layers into DNNs has proven challenging and mostly been limited to single layers [ZHS18] or small datasets [HW19] potentially due to computational and optimisation issues (see, e.g., the discussion in [HW19]). In contrast, in Chapter 6, we show that B-cos networks easily scale to large scale datasets, are easy to optimise, and perform on par with conventional models.

2.4 Model Guidance and Knowledge Distillation

In Chapters 7 and 8, we aim to guide Deep Neural Networks (DNNs) to exhibit certain desirable properties after model training by regularising the explanations for their decisions. Specifically, for this we use human annotations (Chapter 7) or the explanations obtained from a separate teacher model (Chapter 8). In the following, we discuss parallels to prior art and the similarities between the two approaches in detail. In Section 2.4.1, we then provide an additional discussion on relevant works for Chapter 8 specifically.

Implicit Model Guidance via Attribution Priors. Several approaches have been proposed for training better models by enforcing desirable properties on their attributions. These include enforcing consistency under augmentations [PP21, PKO+22, GZF+19], smoothness [EJS+21, MFBS22, KTI19], separation of classes [ZSZ+22, PKO+22, SKL+20, NS20, SMG+20], or constraining the model's attention [FHYF19, AKK+22]. In contrast to these *implicit* attribution priors that are the same for every input, in Chapters 7 and 8, we provide *explicit*, input-specific targets. In particular, we regularise the models' explanations to align with human annotations (Chapter 7) or with explanations of a 'teacher' model (Chapter 8).

Explicit Model Guidance via Attribution Regularization. In contrast to the indirect regularization effect achieved by attribution priors, various approaches have been proposed (cf. [FSSK22, TASD23]) to actively guide models by regularising their attributions, for tasks such as classification [RHDV17, GSZH22, GSB+22, RSMY20, PDN+22, HCMN22, TAvdH20, MFS+21, TK19, Tes19, SST+20, SSS+21, LSES19, SLL+21, YKDO23, FRRLS22], segmentation [LWP+18], VQA [SLS+19, TAvdH20], and knowledge distillation [FTP+22]. The goal here is not only to improve performance, but also make sure that the model is "right for the right reasons" [RHDV17]. This typically involves jointly optimising the models for classification performance, whilst simultaneously ensuring that their explanations localise object features.

In Chapter 7, we conduct an in-depth study to distill the most effective techniques for guiding models via human annotations. Whereas most prior work evaluate on simple datasets [RHDV17, SSS+21, GSZH22, GSB+22] or focuses on one particular attribution method [CSW22], we perform a detailed evaluation across loss functions, attribution methods, models, and 'guidance depths' on challenging real-world multi-label classification datasets. Specifically, we evaluate the localisation losses introduced in the closest related work (RRR [RHDV17], HAICS [SLL+21], and GRADIA [GSZH22]) and additionally propose the EPG metric [WWD+20] as a loss function and show that it has desirable properties.

In Chapter 8, we additionally evaluate whether models can be guided by other DNNs instead of human annotations. Specifically, we propose explanation-enhanced knowledge distillation (e^2KD) and train student models to not only mimic their teacher's logit distribution, but to also yield similar explanations; for this, we use both a popular post-hoc explanation method (GradCAM [SCD+17]) as well as the model-inherent explanations of the B-cos models [BFS22b, BSFS24]. e^2KD thus shares similarities with [PDN+22], in which a model is guided via the attention maps of a vision-language model. Similar to our work in Chapter 8, the authors show that this can guide the students to focus on the 'right' input features. We extend such guidance to KD and discuss the benefits that this yields for KD fidelity.

Evaluating Model Guidance. The benefits of model guidance have typically been shown via improvements in classification performance (e.g. [RHDV17, RSMY20]) or an increase in IoU between object masks and attribution maps (e.g. [GSB+22, LWP+18]). In addition to these metrics, in Chapter 7 we also evaluate on the EPG metric [WWD+20], which has thus far only been used to evaluate the quality of the attribution methods themselves. We further show that it lends itself well to being used as a guidance loss, as it places only minor constraints on the model, and, in contrast to the IoU metric, it is fully differentiable. Moreover, in Chapter 8, we place a particular focus on the benefits of model guidance via a DNN teacher (i.e. of our proposed e^{2}KD) for KD fidelity. For example, as in Chapter 7, we assess whether students learn to be right for the right reasons when training them on highly biased datasets (Waterbirds-100, [SKHL20]). We further evaluate whether e^{2}KD is able to transfer the knowledge more *efficiently* (i.e. with less data) and whether the students are able to learn architectural priors from the teachers.

2.4.1 Explanation-Enhanced Knowledge Distillation (e^{2}KD)

In the following, we additionally discuss relevant works that are specific to our e^{2}KD approach (Chapter 8).

Knowledge Distillation (KD) has been introduced to compress larger models into smaller, more efficient models for cost-effective deployment [HVD15]. Various approaches for KD have since been proposed, which we group into three types for the following discussion: *logit-* [HVD15, ZCS+22, BZR+22], *feature-* [RBK+15, ZK17, SF18, CLZJ21], and *explanation-based KD* [GYLL23, AVT21].

Logit-based KD [HVD15], which optimises the logit distributions of teacher and student to be similar, can suffice to match their accuracies, as long as the models are trained for long enough ('patient teaching') and the models' logits are based on the same images ('consistent teaching'), see [BZR+22]. However, [SIK+21] showed that despite such a careful setup, the function learnt by the student can still significantly differ from the teacher's by comparing the agreement between the two. We expand on [SIK+21] and introduce additional evaluation settings to better evaluate KD fidelity, and show that it can be significantly improved by a surprisingly simple explanation-matching approach. While [OLL22] reports that KD does seem to transfer additional properties to the student, by showing that GradCAM explanations of the students are more similar to the teacher's than those of an independently trained model, we show that explicitly optimising for explanation similarity not only significantly improves this with respect to vanilla KD, but also yields important additional benefits such as a higher robustness to distribution shifts.

Feature-based KD approaches [RBK+15, ZK17, SF18, CLZJ21] provide additional information to the students by optimising some of the students' intermediate activation maps to be similar to those of the teacher. For this, specific choices regarding which layers of teachers and students to match need to be made and these approaches are thus architecture-dependent. In contrast, our proposed e^{2}KD is architecture-agnostic as it matches only the explanations of the models' predictions.

Explanation-based KD approaches have only recently begun to emerge [GYLL23, AVT21] and these are conceptually most related to our work. In CAT-KD [GYLL23], the authors match class activation maps (CAM [ZKL+16]) of students and teachers. As such, CAT-KD can also be considered an 'explanation-enhanced' KD (e^{2}KD) approach. However, the explanation aspect of the CAMs plays only a secondary role in [GYLL23], as the authors even reduce the resolution of the CAMs to 2×2 and fidelity is not considered. In contrast, we explicitly introduce e^{2}KD to improve distillation fidelity and evaluate fidelity across multiple settings. Further, similar to our work, [AVT21] argues that explanations can form part of the model functionality and should be considered in KD. To do so, the authors train an additional autoencoder to mimic the explanations of the teacher model; explanations and predictions are thus produced by separate models. In contrast, we optimise the students directly to yield similar explanations

as the teachers in a simple and parameter-free manner.

Fixed Teaching. Lastly, in Chapter 8 we describe how to distill knowledge via e^{2}KD without querying the teacher at every training step, which we refer to as a 'fixed teacher' setting. This is inspired by [YOH+21, SX22, FPM+23], in which the authors explore pre-computing the logits at the start of training to limit the computational costs due to the teacher. In Chapter 8, we show that this paradigm can seamlessly be extended to e^{2}KD: in addition to pre-computing the teacher's *logits*, we also pre-compute the teacher's *explanations* and show how they can nonetheless be used to guide the student model during distillation.

2.5 Self-Supervised Learning on Long-tail Data

In Chapter 9, we introduce a simple but highly effective approach for improving the quality of the representations learned via contrastive self-supervised representation learning (SSL) on imbalanced data. This approach is based on a novel perspective of the importance of negative samples on the loss function. In the following, we introduce and discuss relevant related works to place our approach into context.

Self-supervised Representation Learning (SSL) from visual data is a quickly evolving field. Recent methods are based on various forms of comparing embeddings between transformations of input images. We divide current methods into two categories: contrastive learning [HFW+20, CFGH20, CKNH20, OLV18], and non-contrastive learning [GSA+20, ZJM+21, CH21, BPL22, WFX+22, GBP+21, ARV19, CMM+20, HCX+22]. Our analysis in Chapter 9 concerns the structure and the properties of the embedding space of contrastive methods when training on imbalanced data. Consequently, this section focuses on contrastive learning methods, their analysis and application to imbalanced training datasets.

Contrastive Learning (CL) employs instance discrimination [WXSL18] to learn representations by forming positive pairs of images through augmentations and a loss formulation that maximises their similarity while simultaneously minimising the similarity to other samples. Methods such as MoCo [HFW+20, CFGH20], SimCLR [CKNH20, CKS+20], SwAV [CMM+20], CPC [OLV18], CMC [TKI20], and Whitening [ESSS21] have shown impressive representation quality and down-stream performance using this learning paradigm. CL has also found applications beyond SSL pre-training, such as multi-modal learning [SCR+22], domain generalisation [YBZ+22], semantic segmentation [VGVGVG21], 3D point cloud understanding [ADD+22], and 3D face generation [DYC+20].

Negatives. The importance of negatives for contrastive learning is remarkable and noticed in many prior works [WWW+21, YHH+21, ZZP+22, ITAC18, KSP+20, RCSJ20, KAG22]. [YHH+21] propose decoupled learning by removing the positive term from the denominator, [RCSJ20] develop an unsupervised hard-negative sampling technique, [WWW+21] propose to employ a triplet loss, and [ZZP+22, KAG22] propose to improve negative mining with the help of different temperatures for positive and negative samples that can be defined as input-independent or input-dependent functions, respectively. In contrast to *explicitly* choosing a specific subset of negatives, we discuss the Info-NCE loss [OLV18] through the lens of an average distance perspective with respect to all negatives and show that the temperature parameter can be used to *implicitly* control the effective number of negatives.

Imbalanced Self-Supervised Learning. Learning on imbalanced data instead of curated balanced datasets is an important application since natural data commonly follows long-tailed distributions [Ree01, LMZ+19, WRH17]. In recent work, [KLX+20], [YX20], [LHGM21], [ZTC+22], [GS22] discover that self-supervised learning generally allows to learn a more robust embedding space than a supervised counterpart. [THvdO21] explore the down-stream performance of contrastive learning on standard

benchmarks based on large-scale uncurated pre-training and propose a multi-stage distillation framework to overcome the shift in the distribution of image classes. [JCMW21, ZYW+22] propose to address the data imbalance by identifying and then emphasising tail samples during training in an unsupervised manner. For this, [JCMW21] compare the outputs of the trained model before and after pruning, assuming that tail samples are more easily 'forgotten' by the pruned model and can thus be identified. [ZYW+22], use the loss value for each input to identify tail samples and then use stronger augmentations for those. Instead of modifying the architecture or the training data of the underlying frameworks, we show that a simple approach—i.e. oscillating the temperature of the Info-NCE loss [OLV18] to alternate between instance and group discrimination—can achieve similar performance improvements at a low cost.

Analysis of Contrastive Learning (CL). Given the success of CL in representation learning, it is essential to understand its properties. While some work analyses the interpretability of embedding spaces [BZK+17, FV18, LFV20, LAV21], here the focus lies on understanding the structure and learning dynamics of the objective function such as in [SPA+19, TWSM20, CLL21]. E.g., [CLL21] study the role of the projection head, the impact of multi-object images, and a feature suppression phenomenon. [WL21b] analyse the feature learning process to understand the role of augmentations in CL. [RSY+21] find that an emphasis on instance discrimination can improve representation of some features at the cost of suppressing otherwise well-learned features. [WI20, WL21a] analyse the uniformity of the representations learned with CL. In particular, [WL21a] focus on the impact of individual negatives and describe a uniformity-tolerance dilemma when choosing the temperature parameter. In Chapter 9, we rely on the previous findings, expand them to long-tailed data distributions and complement them with an understanding of the emergence of semantic structure.

I

THE PROMISES AND PITFALLS OF ATTRIBUTION METHODS

In the first part of this thesis, we aim to highlight the potential benefits as well as discuss the shortcomings of one of the most common ways for explaining Deep Neural Networks (DNNs): importance attribution methods.

For this, **in Chapter 3**, we present a case study that highlights the promise of such methods. Specifically, we apply layer-wise relevance propagation (LRP) to DNNs that were trained to discriminate between brain scans of healthy subjects and patients suffering from Alzheimer's disease (AD). We find that the resulting attribution maps indeed highlight regions that are known to be affected by AD. As such, they could serve as a useful tool for providing clinicians with additional information about the decision of DNNs.

However, while these results are certainly encouraging, it is generally unclear to what degree importance attributions methods can be trusted to faithfully describe the underlying model. This is aggravated by the fact that the importance attributions can drastically vary between different methods and no ground truth to compare against exists.

Therefore, **in Chapter 4**, we present a framework to systematically study and compare different attribution methods. In this context, we also develop an evaluation setting in which, by construction, certain regions are known to not influence the model decisions. Interestingly, we find that some commonly used attribution methods nonetheless attribute importance to those regions and are thus provably not model-faithful. In general, we find that it still remains a challenge to faithfully explain DNNs in a reliable manner and therefore, in the subsequent part of the thesis (Part II), develop novel DNN models that inherently provide faithful and easily interpretable explanations.

3 Explaining MRI-based Alzheimer's Disease Classifiers via LRP

Contents

In the 2018 World Alzheimer Report, it was estimated that 50 million people worldwide were suffering from dementia and this number was projected to rise to more than 152 million people until 2050. The most common reason for dementia is Alzheimer's disease (AD) accounting for around 60-70 % of dementia cases [WHO17]. AD is characterised by abnormal cell death, primarily in the medial temporal lobe. This cell death is thought to be rooted in protein plaques and neurofibrillary tangles, which restrict normal neural function [BES17]. The resulting atrophy is visible in structural magnetic resonance imaging (MRI) data, and derived markers (such as hippocampal volume or grey matter density) have been used to diagnose AD and predict disease progression [FFJJ+10, RHI+17]. In the last decade, those markers have frequently been employed in machine learning settings to allow for predictions on an individual level [KSC+08, OPYM+12, WVA+13, RSW+15, RLW+16]. However, those expert features usually reflect only one part of disease pathology and the combination with standard machine learning methods, such as support vector machines, do not allow for finding new and potentially unexpected hidden data characteristics that might also be important to describe a disease.

By extracting hierarchical information directly from raw or minimally processed data, deep learning approaches can help to fill a gap and offer a great potential for improving automatic disease diagnostics. One family of algorithms that lends itself well to performing non-linear feature extraction from image data and their respective classification into disease categories are convolutional neural networks (CNNs). CNNs have been proven to be very successful in a wide range of medical imaging applications [LKB+17], including AD detection based on neuroimaging data (e.g. [GAM13, SLS14, PM15, ST16, KSBD17]).

Despite this success, automatically learning the features comes at a cost: the decisions of neural networks are notoriously hard to interpret in retrospect. Therefore, deep learning methods including CNNs often face the criticism that they are "black-box" [Cas16]. In contrast to some simpler learning algorithms, in particular decision trees, they do not offer a simple and comprehensible explanation; their architecture is complex and can consist of hundreds of layers with potentially millions of parameters that need to be

trained. In the medical domain, however, it is imperative to base diagnoses and subsequent treatments on an informed decision and not on a single yes / no answer of an algorithm. Therefore, if CNNs are to support clinicians in their daily work, ways have to be found to visualise and interpret their 'decisions'.

A promising technique for doing so is the generation of an individual heatmap for each patient, which lies in the same space as the input image and indicates the importance of each voxel for the classification decision. By allowing for a human-guided, intuitive investigation of what drives the classifier to come to a certain classification decision, such heatmaps hold great potential in assisting and understanding diagnostic decisions performed by DNNs. While a wide range of tools for such importance attributions have been developed (see Chapter 2), for the case study in this chapter we primarily rely on one particular approach for better understanding the DNNs' decisions: layer-wise relevance propagation (LRP, [BBM+15]).

To explain the decision of a DNN, LRP decomposes the network's output score (e.g. for AD) into the individual contributions of the input neurons while keeping the total amount of relevance constant across layers (conservation principle). This has been shown to yield promising results on natural imaging data sets [SBM+16] and performs favourably in comparison to other explanation methods, see also Chapter 4.

In this chapter, we use LRP to explain individual classification decisions for AD patients and healthy controls (HCs) of a CNN trained on structural MRI data (T1-weighted MPRAGE) from the Alzheimer's Disease Neuroimaging Initiative (ADNI[1]). Based on the trained CNN model, we generated LRP heatmaps for each subject in the test set. Importantly, each heatmap indicates the voxel-wise relevance for the particular classification decision (AD or HC). To spot the most relevant regions for AD classification, we computed average heatmaps across AD patients and HCs, which we then further split into correct and wrong classification decisions (i.e. true positives, false positives, true negatives, false negatives). To analyse the relevance in different brain areas according to the Scalable Brain Atlas by Neuromorphometrics Inc. [BTK15], we suggest size-corrected metrics and compared these metrics between LRP and guided backpropagation [SDBR15]. We choose guided backpropagation as a baseline method because (1) gradient-based sensitivity analysis is the most common method for generating heatmaps, (2) it results in more focused heatmaps than using the raw gradients of the models [REW+18], and (3) it is better comparable to LRP than occlusion methods with respect to our relevance measures. On an individual level, we analysed the heatmap patterns of single subjects ('relevance fingerprinting') and correlate them with the hippocampal volume as a key biomarker of AD. We show that the LRP heatmaps succeed in depicting individual contributions to AD diagnosis and thus hold great potential as a diagnostic tool.

Prior to our work [BEWR19] presented in this chapter, the LRP method has been applied to single-trial EEG and functional MRI classification [SLSM16, THMS18], but, to the best of our knowledge, not to clinical disease classification based on structural MRI data. Instead, prior works on explaining the decisions of DNNs on structural MRI data either visualise specific filters and activations [ST16, DSK+19, LPD+18] or perturbation-based explanations (occlusion) [KSBD17, EBPA18, LCWW18]. [REW+18, YRR18] additionally analyse various gradient-based approaches. Similar to our work, prior works qualitatively [KSBD17, EBPA18, LCWW18] and quantitatively [REW+18, YRR18] show that the resulting attributions tend to primarily highlight areas known to be involved in AD (e.g. hippocampus, amygdala or ventricles). However, other areas such as the thalamus or the parietal lobe are also occasionally highlighted, and gradient-based methods tend to be highly distributed [REW+18]. Expanding on prior work, we provide a detailed analysis of LRP as a tool for importance attribution in AD classification, propose additional metrics for evaluating the relative importance assigned to different brain areas, and show that the LRP-based relevance correlates with neurobiological markers such as hippocampal volume.

This chapter is based on [BEWR19] and the corresponding code is publicly available at:
github.com/moboehle/Pytorch-LRP.

[1] http://adni.loni.usc.edu/

3.1 MATERIALS AND METHODS

Data used in the preparation of this work were obtained from the Alzheimer's Disease Neuroimaging Initiative (ADNI, RRID:SCR_003007) database (adni.loni.usc.edu). The ADNI was launched in 2003 as a public-private partnership, led by Principal Investigator Michael W. Weiner, MD. The primary goal of ADNI has been to test whether serial magnetic resonance imaging (MRI), positron emission tomography (PET), other biological markers, and clinical and neuropsychological assessment can be combined to measure the progression of mild cognitive impairment (MCI) and early Alzheimer's disease (AD). For up-to-date information, see www.adni-info.org.
We included structural MRI data of all subjects with Alzheimer's disease (AD) and healthy controls (HCs) listed in the "MRI collection - Standardized 1.5T List - Annual 2 year". The subjects in the data set are labelled as AD if the Clinical Dementia Rating (CDR) score [Mor93] was greater than 0.5. HCs are selected as those subjects with a CDR score of 0. In total, we included 969 individual scans (475 AD, 494 HC) of 193 AD patients and 151 HCs (up to three time points). All scans were acquired with 1.5 T scanners at various sites and had undergone gradient non-linearity, intensity inhomogeneity and phantom-based distortion correction. We downloaded T1-weighted MPRAGE scans and non-linearly registered them to the 1mm resolution 2009c version of the ICBM152 reference brain using Advanced Normalisation Tools (ANTs[2]). This has been done to (1) ensure a relative alignment across subjects, (2) allow the convolutional neural network to extract more robust features, and (3) be able to analyse the heatmaps in a common space. For the region-wise analysis of heatmaps, we used the Scalable Brain Atlas by Neuromorphometrics Inc. [BTK15] available in SPM12[3]. A list of all areas included can be found in the SPM12 package.

3.1.1 Convolutional Neural Network Architecture

Convolutional neural networks (CNNs) are neural networks optimised for array data including images or videos [LBH15]. In addition to input and output layer, they consist of several hidden layers including convolutional and pooling layers. In convolutional layers, in contrast to fully-connected layers, the weights and the bias terms are shared between all neurons in a given layer for a given filter. This means that each of the neurons applies the same *filter* or *kernel* to the input, but at a different position, usually with a displacement (often called stride) of 1-3 between neighbouring neurons. Since these filters are learned via the backpropagation algorithm, CNNs do not rely on hand-crafted features, but can be applied to minimally processed data [LBH15]. CNNs have been very successfully applied to a large number of applications including image and speech recognition [KSH12, AHMJ+14, LSD15] as well as medical imaging and AD classification based on MRI data [GAM13, SLS14, PM15, ST16, KSBD17, LKB+17, VPM17].

The model in the present study consists of four convolutional blocks followed by two fully-connected layers. Each block features a convolutional layer with f filters ($f = 8, 16, 32, 64$) and filter sizes of 3x3x3. Every convolutional layer is followed by batch normalisation and max pooling with window sizes wxwxw ($w = 2, 3, 2, 3$). The fully-connected layers contain 128 and 2 units respectively and dropout ($p = 40\%$) is applied before each. The final fully-connected layer, which is activated by a softmax function serves as the network output, providing the class scores for HCs (first unit) and AD (second unit) respectively. As an optimiser Adam [KB15] was used with an initial learning rate of 0.0001 and a weight decay of 0.0001. The data was split into a training data set (163 AD patients, 121 HCs; 797 images in total), a validation set for optimising the hyperparameters (18 AD patients, 18 HCs; 100 images in total) and a test set (30 AD patients, 30 HCs; 172 images in total). To ensure independence between training and test data, we

[2] http://stnava.github.io/ANTs/
[3] http://www.fil.ion.ucl.ac.uk/spm/software/spm12/

performed the split of the data on the level of patients instead of images. The data was augmented during training by flipping the images along the sagittal axis ($p = 50\%$) and translated along the sagittal axis between -2 and 2 voxels. When the model did not improve for 8 epochs on the validation set, training was stopped. The training epoch (i.e. model checkpoint) with the best validation accuracy (91.00%) was then applied to the test data, resulting in a classification accuracy of 87.96%.

3.1.2 Visualisation Methods

Layer-wise Relevance Propagation (LRP). In the following, we introduce the Layer-wise Relevance Propagation (LRP) algorithm by [BBM+15]. The core idea underlying LRP for attributing relevance to individual input nodes is to trace back contributions to the final output node layer by layer. While several different versions of the LRP algorithm exist, they all share the same principle: the total relevance—e.g. the activation strength of an output node for a certain class—is conserved in each layer; each of the nodes in layer l that contributed to the activation of a node j in the subsequent layer $l+1$ gets attributed a certain share of the relevance R^j_{l+1} of that node. Overall, the sum over the relevances of all nodes i contributing to neuron j in layer l must sum to R^j_{l+1}, such that the total relevance per layer is conserved:

$$\sum_i R^{i\to j}_{l,l+1} = R^j_{l+1} \tag{3.1}$$

For distributing relevance values, different rules have been proposed. Here, we use the β-rule [BML+16]:

$$R^{i\to j}_{l,l+1} = \left((1+\beta)\frac{z^+_{ij}}{z^+_j} - \beta\frac{z^-_{ij}}{z^-_j} \right) R^j_{l+1} \quad . \tag{3.2}$$

Here, $z^{+/-}_{ij}$ refers to the amount of positive/negative input that node i contributed to node j. The individual contributions are divided by the sum over all positive/negative contributions of the nodes in layer l, $z^{+/-}_j = \sum_i z^{+/-}_{ij}$, such that the relevance is conserved from layer $l+1$ to layer l. We have chosen this rule, as it allows for adjusting how much weight is put on positive contributions relative to inhibitory contributions that benefit the AD score. LRP with a β value of zero allows only positive contributions to be shown in the heatmap, whereas non-zero β values additionally correct for the inhibitory effects of neuron activations. When diagnosing AD, the network needs to balance structural evidence speaking for and against AD. Any given local area that looks *healthy* to the network, might have inhibitory effects on the AD score, as it correlates more with HC patients. As the network increases its receptive field size throughout the layers, healthy areas within this receptive field might inhibit the contribution of affected areas to the final class score of AD. By reversing this process with LRP, positive contributions lying closer to healthy areas will thus obtain a lower relevance score, as they overlap with inhibited receptive fields. This leads to sparser heatmaps, see also [BBM+16], and might disproportionately affect small structures surrounded by 'healthy areas'. As AD—especially in the early stages of the disease—can affect brain areas in a highly localised manner, heatmaps obtained with lower β values might thus be more meaningful, as they highlight *all* positive contributions, irrespective of their surroundings. Accordingly, we focus in the present study on $\beta{=}0$, but additionally test the robustness for varying values of β ($\beta{\in}\{0, 0.25, 0.5, 0.75, 1\}$).

For a more detailed description of the LRP algorithm, we kindly refer the reader to [BBM+15, MSM18]. A PyTorch implementation of LRP has been developed for the current work and is available on github.[4]

Guided Backpropagation (GB). In order to emphasise and point out the advantages of LRP as a diagnostic tool, we compared it to a gradient-based method, the guided backpropagation (GB) algorithm

[4] https://www.github.com/moboehle

[SDBR15]. In GB, the absolute values of the gradient of the output with respect to the input nodes is shown as a heatmap, with the additional twist that negative gradients are set to zero at the rectification layers of the network. As was shown by [REW+18], 'rectifying' the gradients in the backward pass leads to more focused heatmaps.

3.1.3 Analysing the Classification Decisions

The CNN model was evaluated on each MR image from the test set and, subsequently, both the LRP and the GB algorithm were used to produce a heatmap for each MR image. For LRP, we produced separate heatmaps for each β value. We analysed the resulting heatmaps (1) group-wise to distill those regions, which are particularly important for the AD classification and (2) individually to understand the network decisions per sample and find differences between subjects. For the former, we computed an average AD heatmap (obtained from all AD subjects) and an average HC heatmap (obtained from all HCs), which we then further split into a true positive heatmap (i.e. average heatmap of clinically validated AD patients, who are classified as AD), a false positive heatmap (i.e. average heatmap of HCs classified as AD), a true negative heatmap (i.e. average heatmap of HCs classified as HC) and a false negative heatmap (i.e. average heatmap of clinically validated AD patients classified as HC). For GB, these heatmaps highlight those areas to which the network is on average most susceptible. For LRP, they show the average relevance of each voxel for contributing to the AD score. All LRP heatmaps show the average relevance for the same class (AD), such that they can be compared on the same scale (relevance for AD diagnosis). As the AD scores of HCs typically range between 0 and 0.5, there will be relevance for AD in HCs, too.

3.1.4 Atlas-based Importance Metrics

To quantitatively analyse the heatmaps and the underlying CNN model, we assessed the importance of different brain areas—as defined by the Neuromorphometrics brain atlas [BTK15]—by using the following three metrics for both LRP and GB.

Sum of AD Importance per Area. As a first metric of importance, the resulting heatmap values were simply summed per area. While this can already be taken as a measure of importance, the resulting importance scores are highly correlated to the area size, see Figure 3.3. Therefore, two size-independent metrics for importance were additionally analysed in more detail: the size-normalised sum, and the average gain (ratio) when comparing to the average HC patient.

Size-normalised AD Importance Metric. For diagnostic purposes, it can be particularly interesting to identify areas that over their entire volume carry a lot of information, i.e. areas with high *relevance density* or, in GB, *susceptibility density*. Hence, we divide the sum of AD importance per area by the size of the area (i.e. number of voxels), which corresponds to the regional mean relevance/susceptibility. While low values over large areas might be due to statistical fluctuations in the data, clusters of relevance (LRP) or susceptibility (GB) in a very confined area could be indicative of the presence of biomarkers for AD.

Gain: Ratio of Values with Respect to the Average HC. Lastly, it is important to note that HCs are not 'relevance-free' under the LRP algorithm: HCs might exhibit certain structural elements in their brains that are correlated with the AD diagnosis. While the network might still classify them as HC, these structures lead to a non-zero class score for virtually every subject. Thus, as an additional metric, we look at the gain in relevance (LRP) and susceptibility (GB) per area, i.e. the ratio to the average HC in that area. By doing this, those areas that differ most between the two cases will be attributed the highest importance.

3.2 RESULTS

In Section 3.2.1, we compare the heatmaps generated by GB and LRP qualitatively with respect to different β values and different sets of data (AD, HC, true positives, false positives etc., see Figure 3.1-3.2). In Section 3.2.2, we quantitatively compare the heatmaps with respect to the different atlas-based importance metrics (see Figure 3.3-3.6). In Section 3.2.3, we present and discuss the LRP heatmaps of two individual patients (see Figure 3.7) and investigate the association between LRP relevance scores and hippocampal volume as one of the neurobiological key markers of AD (see Figure 3.8).

3.2.1 Average Heatmap Comparison

In Figure 3.1, we show the average heatmaps for AD patients and HCs, separately for LRP with different β values ($\beta = 0, 0.5, 1$) and GB. The AD pattern between LRP and GB is relatively similar, which is reasonable because all heatmaps are extracted from the same CNN model. However, whereas GB heatmaps are very susceptible for both AD and HCs, LRP heatmaps show much more relevance in AD patients than HCs. This indicates that LRP heatmaps might be more valuable in assessing why a certain person has been classified as AD patient as opposed to which voxels should be changed to increase the likelihood for AD diagnosis. Concerning the different β values, it is noted that the heatmaps look qualitatively similar, but that sparseness increases with higher β values (which is due to a larger effect of inhibitory contributions, see also [BBM+16]). Since β values close to 0 focus on positive AD contributions and might thus be clinically more relevant, we focus on $\beta = 0$ in the remaining analyses.

In Figure 3.2, we show the average heatmaps for the distinct classification cases (true positives, false positives etc.), separately for LRP ($\beta = 0$) and GB. In particular, the false positives lead to an interesting insight: For LRP, the false positives exhibit less relevance than the true positives, but generally in similar areas. This could indicate that in these patients structures that are correlated with AD were found, albeit that overall the positive contribution was less compelling than for true AD patients. For GB on the other hand, the false classifications (mostly false positives, but also false negatives) seem to exhibit the highest gradient values of all cases. This exemplifies well what GB truly measures: in the case of false positives (and negatives), the network might be 'unsure' and more easily influenced to change its decision; the outcome is unstable. The highlighted areas that could change the outcome are very broadly distributed and need not necessarily represent areas with positive contributions for AD.

3.2.2 Atlas-based Importance Metrics

In Figure 3.3, we show the sum of AD importance per area, separately for LRP ($\beta = 0$) and GB. Although this metric seems to be dominated by the size of the respective brain area, one important qualitative difference between LRP and GB is visible: in the LRP results, the mean importance values per area are consistently much higher for AD patients than for HCs. For GB, this clear split is not present; moreover, the average sum of gradients in several brain regions, including the cerebral white matter and cerebellum, is even higher for HC than for AD. This exemplifies well that the heatmaps for GB cannot directly be interpreted as showing the relevance for AD classification, but instead show the sensitivity to certain areas, which does not have to be AD or HC specific. As the absolute sum of importance correlates with the size of the respective brain area, in the following, we additionally control for the size of the brain area.

In Figure 3.4, the total sum of importance is normalised by the size of the respective brain area. Here, the aforementioned difference in the distributions between HCs and AD patients becomes even more apparent: while the distributions are very heavily overlapping for GB, this is not the case for LRP. Notably, the

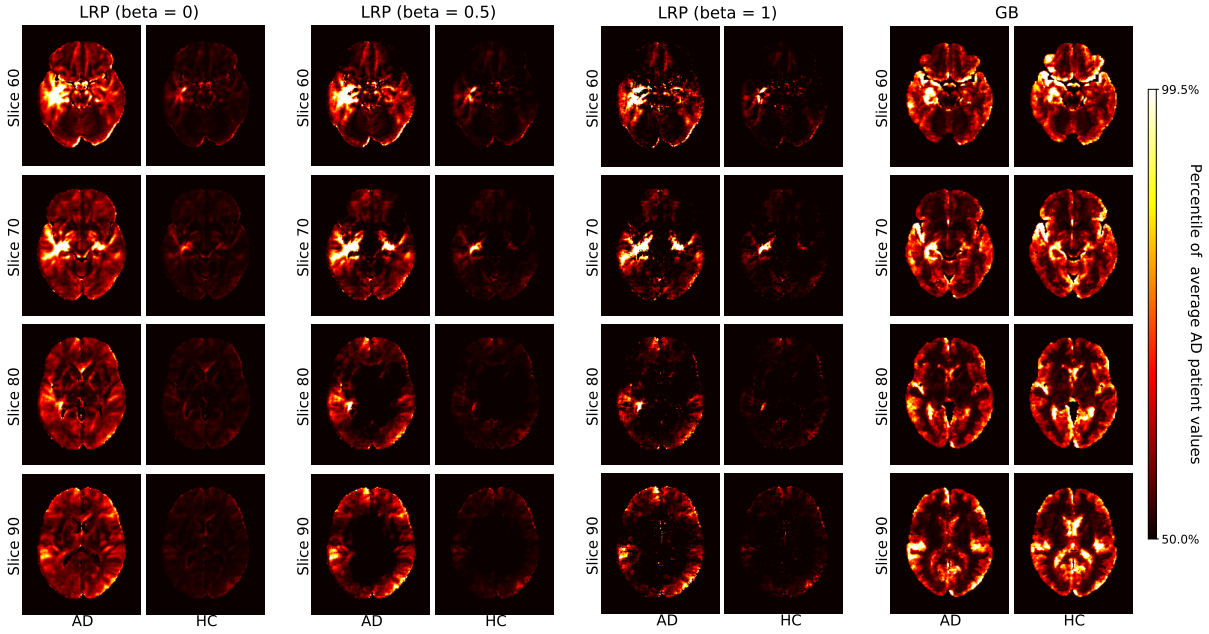


Figure 3.1: Average heatmaps for AD patients and healthy controls (HCs) in the test set are shown separately for LRP with $\beta = 0, 0.5, 1$ (left) and GB (right). The scale for the heatmap is chosen relative to the average AD patient heatmap for LRP and GB respectively. Hence, values in the average heatmaps that are higher than the 50th percentile and lower than the 99.5th percentile are linearly colour-coded as shown on the scale. Values below (above) these numbers are black (white).

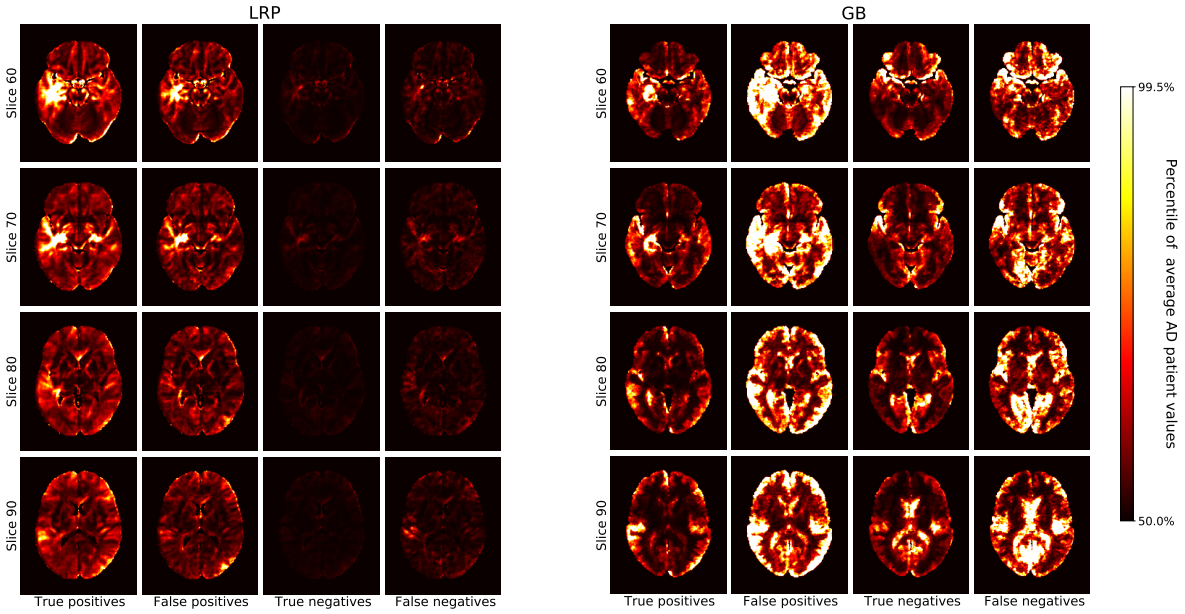


Figure 3.2: The average heatmaps over all subjects in the test set are plotted for the following cases (left to right): true positives, false positives, true negatives, and false negatives; separately for LRP with $\beta = 0$ (left) and GB (right). For each heatmap, the colour-coding is the same as in Figure 3.1, i.e. with all values smaller than the 50th percentile of the average AD patient in black, increasing values going over red to yellow, and all values greater than the 99.5th percentile in white.

variance in the AD distributions is much higher in the AD case than in the HC case. This could indicate that the network has learned to differentiate between subtypes of AD and bases its decision on different structural elements for different patients; the existence of different subtypes of AD has been investigated in recent work, see for example [PNK+17, FVHC+17]. In contrast, for HCs the relevance density is consistently very low. As an example of the diversity in importance assessments according to this metric, we added the 'individual fingerprints' of two AD patients to Figure 3.4; for these patients the individual heatmaps will be compared in Section 3.2.3 and Figure 3.7. In Figure 3.5, the results for the gain metric for different cases—true positives and true negatives—are visualised. This metric allows for plotting the LRP and the GB results on the same scale and emphasises once again the stronger distinction between AD patients and HCs under the LRP algorithm. Most gain for LRP has been found in areas of temporal lobe including transversal temporal gyrus, hippocampus, planum temporale and amygdala.

In Figure 3.6, we compare the regional overlap of the top 10 regions between different β values ($\beta \in \{0, 0.25, 0.5, 0.75, 1\}$), separately for the three metrics. It can be seen that (1) the regional overlap is strongest for relevance sum followed by relevance density and relatively unstable for gain of relevance especially for large and more distant β values and (2) the regional overlap is stronger for neighbouring β values. The instability of the gain metric for higher β values is probably due to the associated sparsity leading to very low relevance scores for HCs (which might thus in some cases inflate the gain metric).

3.2.3 Individual Heatmaps – Fingerprinting and Neurobiological Relevance

Since the LRP heatmaps take into account the individual filter activations and therefore highlight positive contributions to the class score of AD, they might serve as 'individual fingerprints' in a diagnostic tool. In Figure 3.7, we show several slices of the relevance heatmaps for two patients in order to highlight the diversity in those heatmaps. The two patients were selected from the test set as those with the highest cosine distance in the relevance-density space between each other among those patients that were classified as AD with a class score $> 90\%$ (their individual trajectories of region-wise relevance are shown in Figure 3.4). It can be seen that the areas, which mainly contributed to the network decision, are rather different for the two patients. For one patient (patient B), the class score of the network is heavily influenced by areas of the temporal lobe, such as parahippocampal gyrus, entorhinal area, hippocampus, inferior temporal gyrus and amygdala, while for the second patient (patient A), frontal areas, including triangular part of inferior frontal gyrus, superior frontal gyrus and frontal pole, in addition to superior temporal gyrus seem to be most informative.

To investigate whether higher importance scores correspond to stronger anatomical deviations (e.g. atrophy) in correctly classified AD patients (true positives), we performed a correlation analysis between hippocampal volume and LRP relevance / GB susceptibility scores (see Figure 3.8). We show that the LRP relevance score ($\beta = 0$) in the hippocampus is significantly (negatively) correlated with hippocampal volume ($-0.560, p < 10^{-3}$, permutation test), whereas the GB score is not ($0.096, p = 0.77$). To rule out that false positives are outliers in terms of association between hippocampal volume and LRP relevance, we included them in Figure 3.8. Interestingly, for larger β values the correlation tends to decrease ($-0.560, -0.562, -0.525, -0.457, -0.361$ for $\beta = 0, 0.25, 0.5, 0.75, 1$ respectively) supporting our notion of a higher neurobiological relevance in case of β values close to 0.

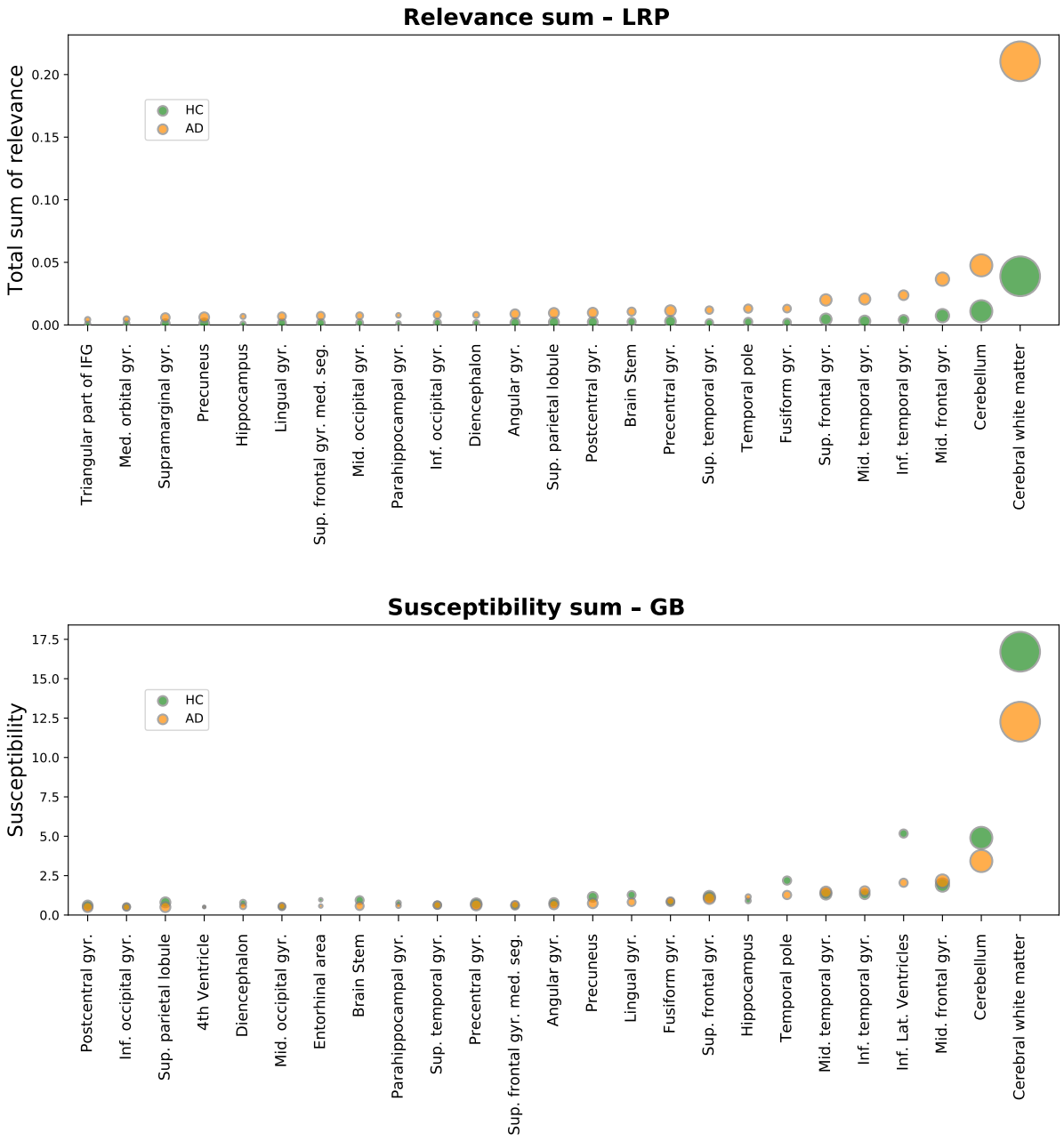


Figure 3.3: Absolute sum of relevance (LRP, top) and absolute sum of susceptibility (GB, bottom) is shown for different brain areas. Susceptibility refers to the absolute value of the GB gradients. Only the top 25 most important areas under this metric are shown for LRP and GB respectively. The circles show the average sum for each area over all AD patients (orange) and all healthy controls (HCs, green) in the test set. Note that the absolute sum metric shows significant correlations with the size of the respective brain areas.

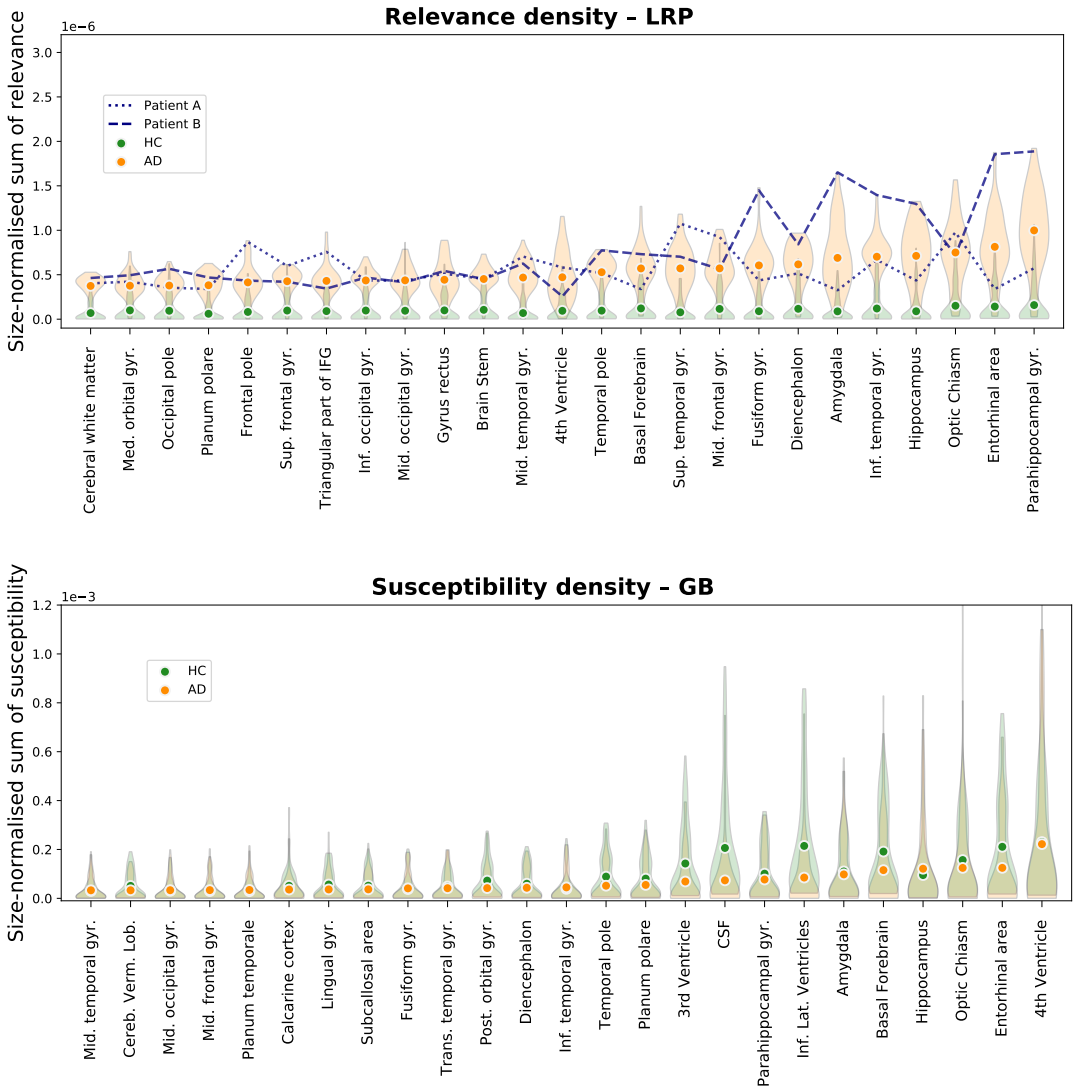


Figure 3.4: Size-normalised relevance (LRP, top) and size-normalised susceptibility (GB, bottom) is shown for different brain areas. Only the top 25 most important areas under this metric are shown for LRP and GB respectively. We show the average density for all AD patients (orange circles) and all healthy controls (HCs, green circles) in the test set along with a density estimation of the distribution of values per area (orange and green shaded area for AD and HCs respectively). Moreover, two patients were selected to emphasise the diversity in relevance distributions for LRP; the patients were selected as those with the highest cosine distance in the relevance-density space of the 25 areas between each other among those patients that were classified as AD with a class score >90%.

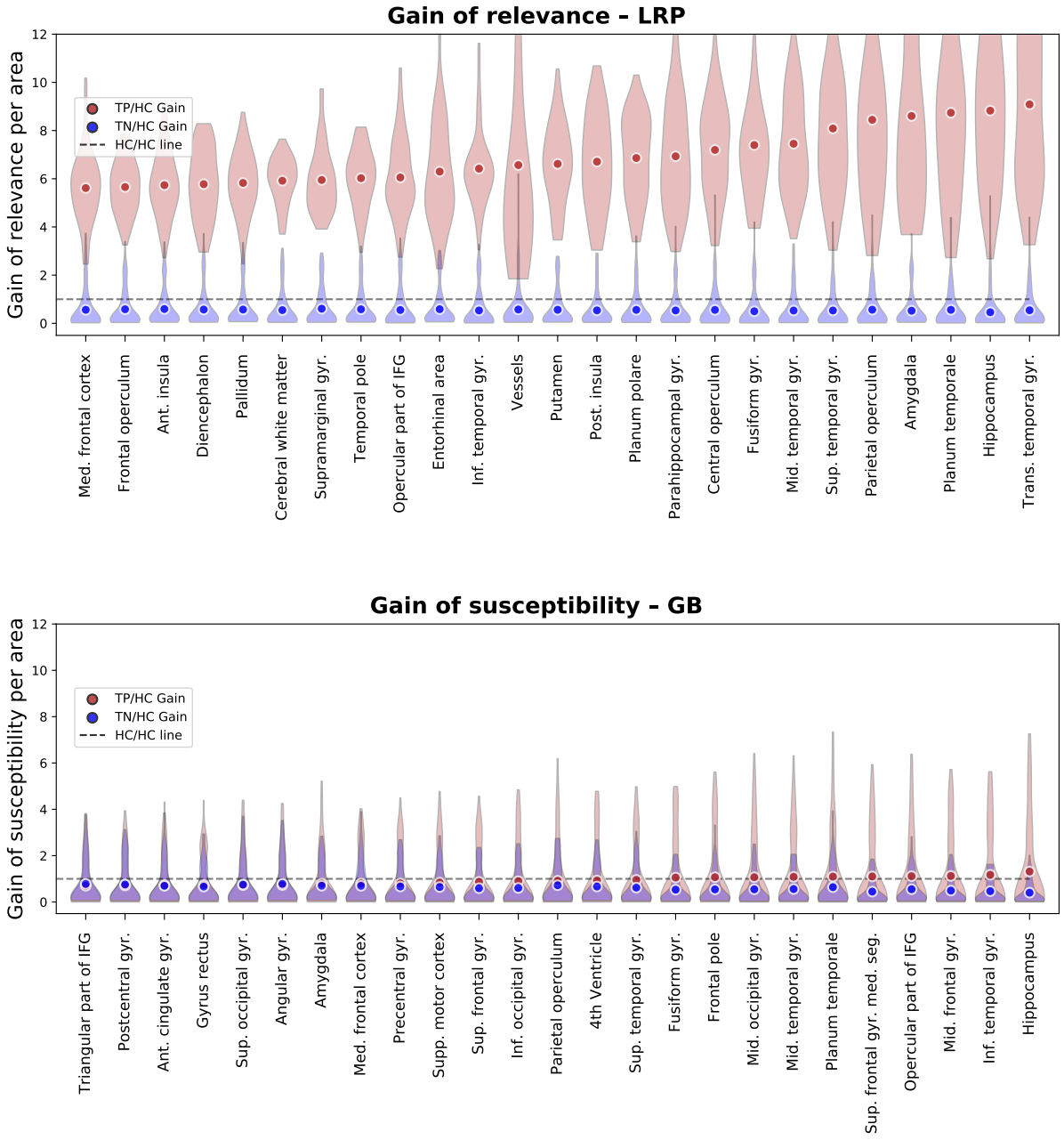


Figure 3.5: Gain of relevance (LRP, top) and gain of susceptibility (GB, bottom) is shown for different brain areas. The gain per area is defined as the average sum of relevance (LRP) or susceptibility (GB) in a given area divided by the average sum in this area over all healthy controls (HCs) in the test set. Again, only the top 25 most important areas under this metric are shown for LRP and GB respectively. To provide an estimate of gain in correctly classified subjects, we show here the mean and density estimations only for true positive (TP) and true negative (TN) classifications. As an additional visual aid, the identity gain (gain of 1) is shown as a dashed line.

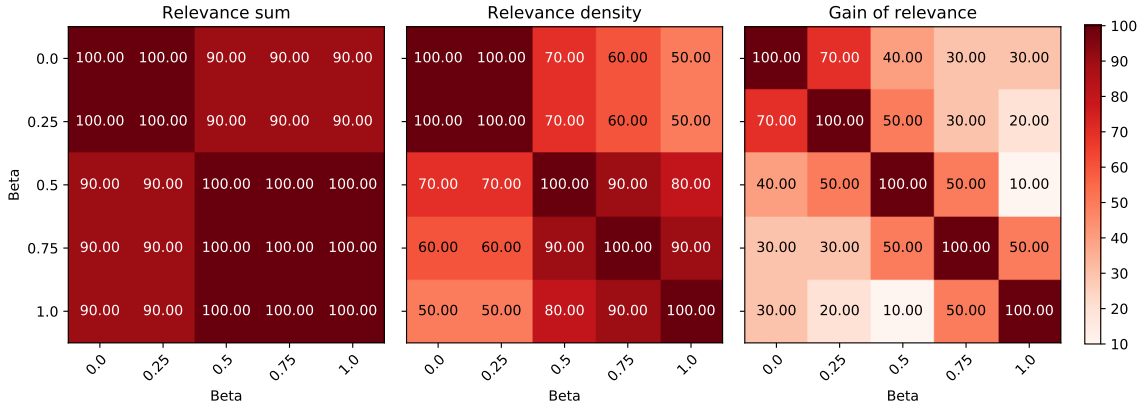


Figure 3.6: Comparison of the effect of different β values on the regional ordering in Figures 3.3 to 3.5. The intersection between the top 10 regions of the three metrics is shown for different LRP β values in %.

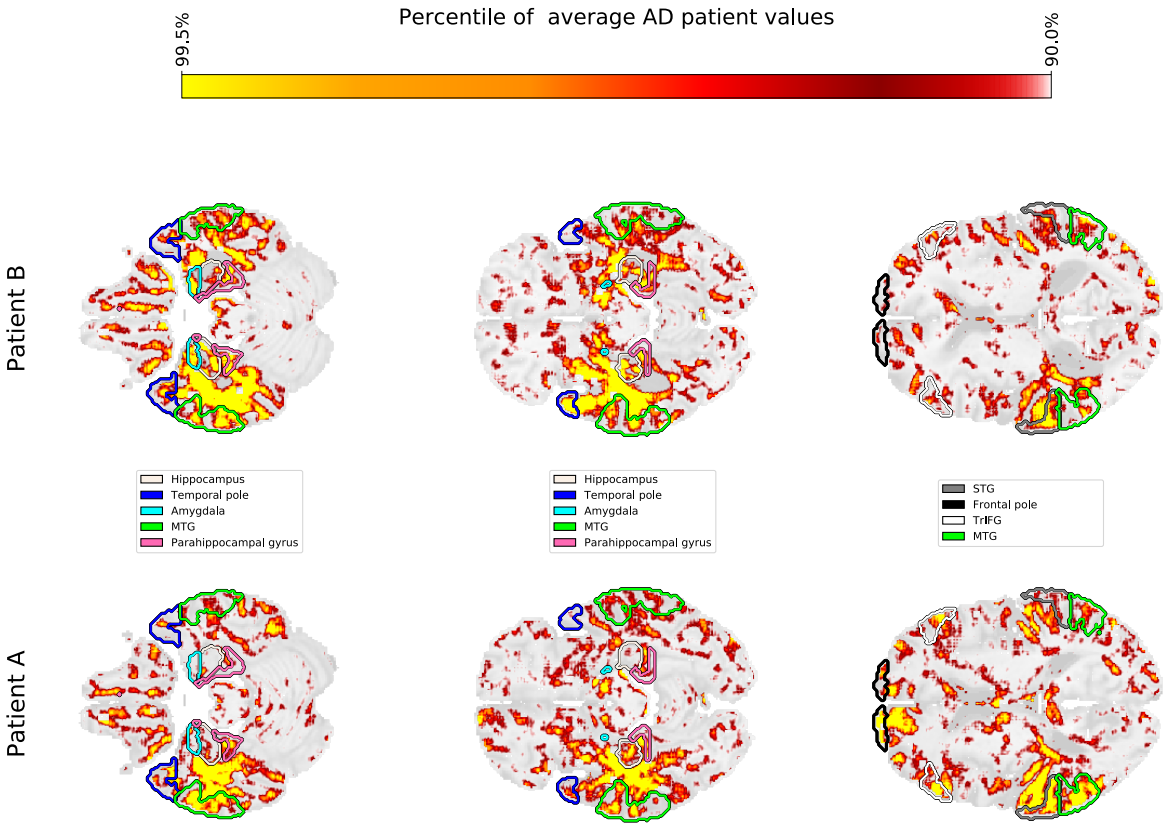


Figure 3.7: Three brain slices are shown for patient A and patient B, whose individual slopes in relevance density have been shown in Figure 3.4. The highlighted areas are the hippocampus, temporal pole, amygdala, parahippocampal gyrus, medial temporal gyrus (MTG), superior temporal gyrus (STG), triangular part of the inferior frontal gyrus (TrIFG) and frontal pole. The scale for the heatmap is chosen relative to the average AD patient heatmap. Hence, values in the individual patients that are higher than the 90th percentile and lower than the 99.5th percentile are linearly colour-coded as shown on the scale. Values below (above) these numbers are transparent (yellow).

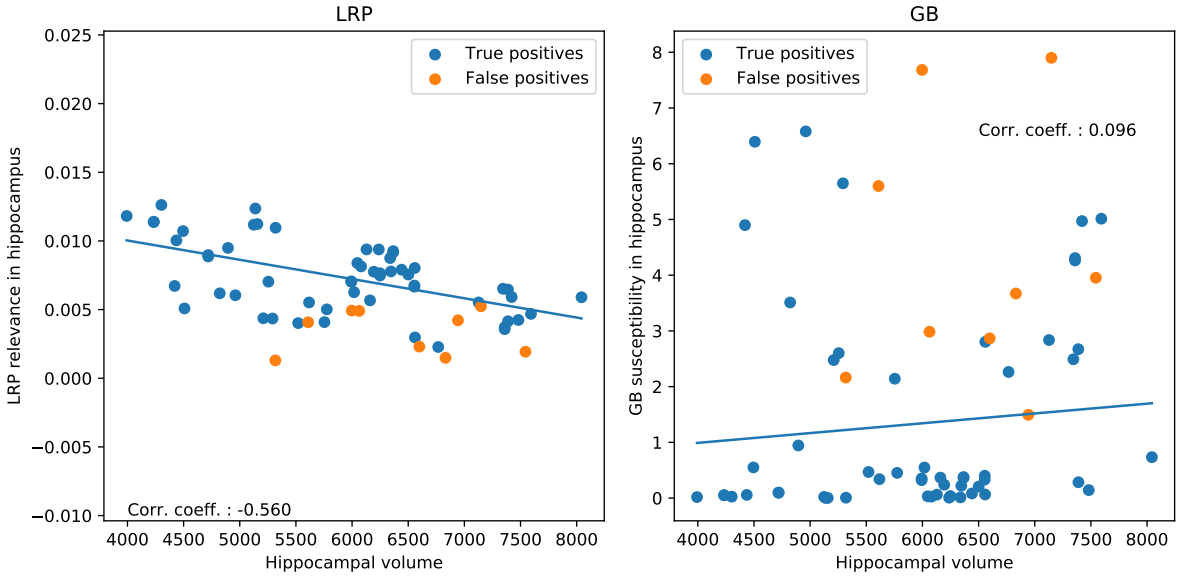


Figure 3.8: Correlation between hippocampal volume and LRP relevance / GB susceptibility in hippocampus for correctly classified AD patients (true positives; left: LRP, right: GB), as well as the false positives.

3.3 DISCUSSION

In this study, we studied LRP as a method for explaining individual CNN decisions in AD classification. After training a CNN to separate AD patients and HCs based on structural MRI data, individual heatmaps—indicating the importance for each voxel for the respective classification decision—were produced for the test subjects. We analysed the heatmaps with respect to different classification subgroups (AD patients, HCs, true positives, false positives etc.) and different β values. The relevance of brain regions contained in the Neuromorphometrics atlas was evaluated using three different importance metrics, namely the sum of importance per area, the size-normalised AD importance, and the gain as ratio between AD and HC importance. We demonstrated that LRP-derived heatmaps—in contrast to GB—provide (1) high specificity for individuals and (2) little relevance for AD in HCs. Additionally, areas that exhibit a lot of relevance correlate well with what is known from literature. Importantly, these LRP heatmaps were produced without the need for expert annotations on the presence or absence of biomarkers throughout the learning process. This combination of a simple classification task (AD vs. HC) and in-depth network analysis by LRP might be a promising tool for diagnostics. Additionally, it could allow for discovering new and unknown biomarkers for a variety of diseases and might help distinguishing subtypes of AD by analysing the diversity in 'relevance hot-spots' across all AD patients. Further, the size-corrected metrics ('relevance density' and 'relevance gain') correlate with what is known from AD research, indicating that the most discriminating features for classifying an input image as AD can be found in the temporal lobe.

3.3.1 Regional Specificity of LRP

We quantitatively evaluated the heatmaps, obtained by either GB or LRP, towards different brain areas according to the Neuromorphometrics atlas [BTK15] by summarising the importance (AD relevance in case of LRP, susceptibility in case of GB) for each brain area separately. Both types of heatmaps mostly identified regions known to be important in disease progression of AD, such as structures in the medial temporal lobe including hippocampus, amygdala, parahippocampal gyrus, and entorhinal cortex [DSA+01, DCH+09, FFJJ+10, WVA+13, VPW+13, KKHR+14, LCJ+17] as well as frontal and parietal areas [CWW+11, QSR+13, KHF+17, PNK+17, LCWW18]. For all these regions morphometric changes including global and local atrophy (e.g. smaller volumes of hippocampus or amygdala, reduced cortical thickness or grey matter density) or deviations in shape have been shown and related to disease progression and cognitive decline [DCH+09, FFJJ+10, WVA+13, HMRGP14, LCJ+17, LSG+18]. These changes seem to be utilised by our CNN framework for making individual predictions and are highlighted in the heatmaps of both LRP and GB. However, the contrast in importance scores between AD patients and HCs is much higher for LRP than GB (in GB, the average heatmaps for AD patients and HCs are quite similar). This supports the notion that LRP heatmaps reflect AD-specific relevance, whereas GB emphasises areas which the network more generally is sensitive to. Regarding other structures found to be important in our network, it might be interesting to see if also other neural networks find relevance in these areas and if predictions about finding significant structural changes in these areas might be possible at some point. In this respect, the decisions of such networks can be treated as a 'second opinion' and a reciprocal learning process with medical experts might be initiated.

3.3.2 Fingerprinting and Neurobiological Relevance

In addition to heatmap differences between AD patients and HCs, we noticed a high variability between the heatmaps of individual AD patients for the LRP method. This variability was not only reflected in a high variance of important scores within regions, but also in individual trajectories ('fingerprints'), which

we exemplary depicted for two AD patients, see Figure 3.7. For future work, it might be very interesting to see if these fingerprints reflect different disease stages of AD [BB91, CWW+11] or allow for identifying subtypes of AD, in which brain areas are affected differently [MGRR+11, NJL+14, SGGP+16, ZMS+16, FVHC+17, PNK+17]. [ZMS+16], for example, identified a temporal, a subcortical and a cortical atrophy factor associated with impairment in different cognitive domains. Another important question is whether the relevance found by the LRP method reflect some true evidence in the sense of biomarkers. By showing that the hippocampal volume is significantly correlated to the LRP relevance scores (but not to the GB susceptibility scores), we argue that LRP—at least partially—succeeded here in breaking down the relevance to the level of voxels in a meaningful way. Interestingly, we found higher correlations for lower β values speaking for a higher neurobiological relevance of β values close to 0. Further studies are needed to more carefully relate LRP relevance measures to other clinical markers of AD including biomarkers and neuropsychological test scores, also in dependency of different CNN models and parameter settings. Moreover, our metrics should be evaluated in patients with mild cognitive impairment (MCI).

3.3.3 Limitations

Although LRP heatmaps seem to be a promising tool for visualising neural network decisions, we would like to point out several limitations of LRP and other heatmap methods in the context of this study.

First, heatmap methods are limited by the lack of a ground truth. Most commonly, heatmaps are qualitatively evaluated based on visual assessment, but there are also studies proposing sanity checks [AGM+18] or more objective quality measures such as region perturbation [SBM+16]. In [Lip18], the interpretability of models has been generally investigated and questioned. In medical research, heatmaps can be qualitatively evaluated based on prior knowledge (e.g. hippocampus is known to be strongly affected in AD, therefore it seems reasonable to find relevance there). Given that in the specific case of heatmaps for MR images the input space is highly structured, we proposed here additional ways for assessing the quality of explanations by using a brain atlas. Future studies might assess the neurobiological validity by removing presumably important brain areas and re-training the classifier.

Second, it is largely acknowledged that heatmaps are quite sensitive to the specific algorithms (and its parameters, e.g. the β value in case of LRP) used to produce them. However, regarding the β values in LRP, we have shown that the heatmaps are relatively robust towards this parameter, only sparsity increases as a function of β. Additionally, we demonstrated that the regional ordering is relatively stable for relevance sum and density, but unstable for the gain metric—especially in the case of large and more distant β values.

Third, heatmaps just highlight voxels that contributed to a certain classifier decision, but do not allow making a statement about the underlying reasons (e.g. atrophy or shape differences) or potential interactions between voxels or brain areas. For example, it is difficult to disentangle interactions between different regions (certain patterns in the hippocampus might only be considered as positive evidence if structure Y is found in area Z) nor do we know whether the network developed specific filters for atrophy or the shapes of different structures. Although we found in this study a significant correlation between hippocampal volume and LRP relevance measures, we can not make any claim about causal relationships here. Future studies are necessary to more systematically investigate the relationship between manifested neurobiological markers and LRP explanations.

Fourth, heatmaps strongly depend on the type and quality of the classifier, whose decisions are sought to be explained. Therefore, each heatmap should be read as an indication of where the specific network model sees evidence. For badly trained networks, this does not have to correlate at all with the presence of actual biomarkers. Nevertheless, the better the classifier, the more likely it becomes that the classifier

uses meaningful patterns as a basis for its decision and that the heatmaps correlate with 'true' evidence for AD. However, heatmaps are also useful in cases, where classification performance is low or sample size is rather small, e.g. for better understanding if the classifier picks up relevant or irrelevant features (e.g. noise or imaging artifacts) and if there are any biases present in the data set [LBM+16, MSM18]. It would be very interesting to investigate how the heatmaps change for different networks, as those which yield stronger classification results should also base their decisions on *better* 'evidence'.

And finally, we point out that when referring to brain areas throughout this work, we refer to the location that the areas are assigned in the brain atlas and not to the individual anatomical structures of any patient. Due to inter-individual differences, the individual patient's anatomical realities will not perfectly match the atlas; this is most likely further aggravated by the presence of atrophy in AD patients.

3.4 CONCLUSION

In conclusion, we introduced the LRP method for explaining individual CNN decisions in MRI-based AD diagnosis. In contrast to GB, LRP heatmaps can be interpreted as providing individual AD relevance ("What speaks for AD in this particular subject?") as opposed to a general susceptibility for small variations in the input data. Additionally, we provided a framework and specific metrics (i.e. 'relevance density' and 'relevance gain') to quantitatively compare heatmaps between different groups, brain areas or methods. We demonstrated that these metrics correlate well with clinical findings in AD, but also vary strongly between AD patients. By this, the LRP method might be very useful in a clinical setting for a case-by-case evaluation. However, we would like to point out that (1) our metrics should be evaluated in different network architectures and (2) other (individual) brain atlases might be used for the evaluation of regions. Future studies should evaluate the LRP method on patients with mild-cognitive impairment (MCI) and relate findings to known biomarkers in AD. We are convinced that our framework might also be very useful for other disease classification studies in helping to understand individual network decisions.

4 Systematic Evaluations of Attribution Methods

Contents

Apart from the LRP framework discussed in Chapter 3, many other techniques have been developed for attributing importance values to individual input features and to thereby shed light on the decision-making process within Deep Neural Networks (DNNs). However, as discussed in the previous chapter, the lack of access to a ground truth for those importance values makes it difficult to reliably compare and evaluate these attribution methods in a holistic and systematic manner.

To better understand advantages and shortcomings of different attribution methods, in this chapter we develop a systematic framework for comparing them. For this, we place a particular focus on three important components: (1) reliably measuring the attributions' *faithfulness*, (2), ensuring a *fair comparison*, and (3) providing a framework that allows for *systematic* visual inspections of their attributions.

To address (1), we introduce a novel evaluation scheme (**DiFull**), which allows distinguishing possible from impossible importance attributions. This effectively provides ground truth annotations for whether or not an input feature can possibly have influenced the model output. As such, it can highlight distinct failure modes of attribution methods (Figure 4.1, left).

For (2), we argue that a fair evaluation requires attribution methods to be compared on equal footing. However, we observe that different methods are often used to explain DNNs to different depths, thus effectively solving different problems (e.g., explaining the full model vs. the classification head). To even the playing field, we propose a multi-layer evaluation scheme for attributions (**ML-Att**) and thoroughly evaluate commonly used methods across multiple layers and models (Figure 4.1, left). When compared on the same level, we find that performance differences between some methods essentially vanish.

Finally, for (3), we note that relying on individual examples for a qualitative comparison is prone to skew the comparison and cannot fully represent the evaluated attribution methods. To overcome this, we propose a qualitative evaluation scheme for which we aggregate attribution maps (**AggAtt**) across many input samples. This allows us to observe trends in the performance of attribution methods across complete datasets, in addition to looking at individual examples (Figure 4.1, right).

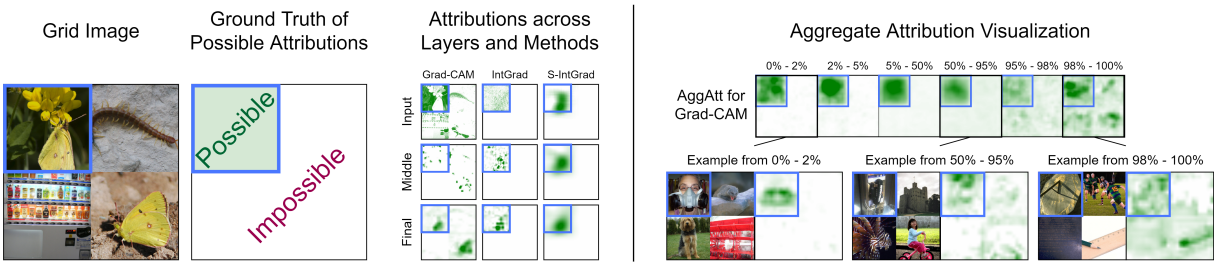


Figure 4.1: **Left:** Illustration of *DiFull* and *ML-Att*. In DiFull, we evaluate models on image grids (col. 1). Crucially, we employ separate *classification heads* for each subimage that cannot possibly be influenced by other subimages; this yields 'ground truths' for possible and impossible attributions (col. 2). For ML-Att, we evaluate methods at different network layers; and show attributions for the example grid image using GradCAM and IntGrad. Further, we show results after smoothing IntGrad (S-IntGrad), which we find to perform well (Section 4.3.3). GradCAM, for instance, incorrectly attributes the bottom-right butterfly which lies in the 'impossible' partition for attributions. **Right:** Visualisation of our AggAtt evaluation. By sorting attributions into percentile ranges w.r.t. their performance and aggregating them over many samples, we obtain a holistic view of a methods' performance. AggAtt can thus reflect both best and worst case behaviour of an attribution method.

In short, we would like to highlight **the following contributions:**

(1) We propose a novel evaluation setting, **DiFull**, in which we control which regions *cannot possibly* influence a model's output, which allows us to highlight definite failure modes of attribution methods.

(2) We argue that methods can only be compared fairly when evaluated *on the same layer*. To do this, we introduce **ML-Att** and evaluate all attribution methods at multiple layers. We show that, when compared fairly, apparent performance differences between some methods effectively vanish.

(3) We propose a novel aggregation method, **AggAtt**, to qualitatively evaluate attribution methods across all images in a dataset. This allows to qualitatively assess a method's performance across many samples (Figure 4.1, right), which complements the evaluation on individual samples.

(4) We apply our proposed qualitative and quantitative evaluations to a wide range of importance attribution methods and show that the results and trends are consistent across a diverse set of CNN architectures. In this context, we find that the LRP approach evaluated in Chapter 3 performs favourably in comparison to other methods. However, we also observe that achieving good localisation requires carefully choosing propagation rules and their parameters, and that LRP is not implementation invariant.

(5) We propose a post-processing smoothing step that significantly improves localisation performance on some attribution methods. We observe significant differences when evaluating these smoothed attributions on different architectures, which highlights how architectural design choices can influence an attribution method's applicability.

This chapter is based on [RBS22] and [RBS24] and the corresponding code is publicly available at: github.com/sukrutrao/Attribution-Evaluation.

4.1 EVALUATING ATTRIBUTION METHODS

We present our evaluation settings for better understanding the strengths and shortcomings of attribution methods. Similar to the Grid Pointing Game (GridPG) [BFS21], these metrics evaluate attribution methods on image grids with multiple classes. In particular, we propose a novel quantitative metric, DiFull, and an extension to it, DiPart (4.1.1), as stricter tests of model faithfulness than GridPG. Further, we present a qualitative metric, AggAtt (4.1.2) and an evaluation setting that compares methods at identical layers, ML-Att (4.1.3).

4.1.1 Quantitative Evaluation: Disconnecting Inputs

In the following, we introduce the quantitative metrics that we use to compare attribution methods. For this, we first describe GridPG and the grid dataset construction it uses [BFS21]. We then devise a novel setting, in which we carefully control which features can influence the model output. By construction, this provides ground truth annotations for image regions that can or cannot possibly have influenced the model output. While GridPG evaluates how well the methods localise class discriminative features, our metrics complement it by evaluating their *model-faithfulness*.

4.1.1.1 *Grid Data and GridPG*

For GridPG [BFS21], the attribution methods are evaluated on a synthetic grid of $n \times n$ images in which each class may occur at most once. In particular, for each of the occurring classes, GridPG measures the fraction of positive attribution assigned to the respective grid cell versus the overall amount of positive attribution. Specifically, let $A^+(p)$ refer to the positive attribution given to the p^{th} pixel. The localisation score for the subimage x_i is given by:

$$L_i = \frac{\sum_{p \in x_i} A^+(p)}{\sum_{j=1}^{n^2} \sum_{p \in x_j} A^+(p)} \tag{4.1}$$

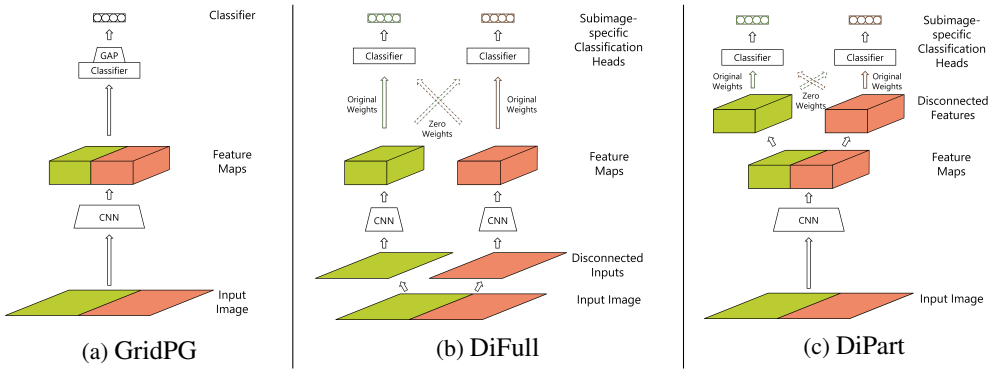
An 'optimal' attribution map would thus yield $L_i = 1$, while uniformly distributing attributions would yield $L_i = \frac{1}{n^2}$.

By only using confidently classified images from distinct classes, GridPG aims to ensure that the model does not find 'positive evidence' for any of the occurring classes in the grid cells of other classes. However, specifically for class-combinations that share low-level features, this assumption might not hold, see Figure 4.3 (right): despite the two dogs (upper left and lower right) being classified correctly as single images, the output for the logit of the dog in the upper left is influenced by the features of the dog in the lower right in the grid image. Since all images in the grid can indeed influence the model output in GridPG[1], it is unclear whether such an attribution is in fact not *model-faithful*.

4.1.1.2 *Proposed Metric: DiFull*

As discussed, the assumption in GridPG that no feature outside the subimage of a given class should positively influence the respective class logit might not hold. Hence, we propose to fully disconnect (DiFull) the individual subimages from the model outputs for other classes. For this, we introduce two modifications. First, after removing the GAP operation, we use $n \times n$ classification heads, one for

[1] As shown in Figure 4.2a, the convolutional layers of the model under consideration process the entire grid to obtain feature maps, which are then classified point-wise. Finally, a *single output per class* is obtained by globally pooling all point-wise classification scores. As such, the class logits can, of course, be influenced by all images in the grid.



(a) GridPG (b) DiFull (c) DiPart

Figure 4.2: **Our three evaluation settings.** In GridPG, the classification scores are influenced by the entire input. In DiFull, on the other hand, we explicitly control which inputs can influence the classification score. For this, we pass each subimage separately through the spatial layers, and then construct individual classification heads for each of the subimages. DiPart serves as a more natural setting to DiFull, that still provides partial control over information. We show a 1×2 grid for readability, but the experiments use 2×2 grids.

each subimage, and only *locally* pool those outputs that have their receptive field centre *above the same subimage*. Second, we ensure that their receptive field does not overlap with other subimages by zeroing out the respective connections.

In particular, we implement DiFull by passing the subimages separately through the CNN backbone of the model under consideration[2], see Figure 4.2b. Then, we apply the classification head separately to the feature maps of each subimage. As we discuss in Appendix A, DiFull has similar computational requirements as GridPG.

As a result, we can *guarantee* that no feature outside the subimage of a given class can possibly have influenced the respective class logit—they are indeed *fully disconnected*.

Note that this setting differs from pixel removal metrics (e.g. [SBM+16, SF19]), where 'removing' a patch of pixels at the input and replacing it with a baseline (e.g. zero) values may still result in the patch influencing the network's decision, for example, based on the shape and the location of the patch. In contrast, we effectively make the *weights* between the CNN backbone and the classification heads for other grid cells zero, which ensures no influence from pixels in those grid cells to the output.

4.1.1.3 *Natural Extension: DiPart*

At one end, GridPG allows any subimage to influence the output for any other class, while at the other, DiFull completely disconnects the subimages. In contrast to GridPG, DiFull might be seen as a constructed setting not seen in typical networks. As a more natural setting, we therefore propose DiPart, for which we only partially disconnect the subimages from the outputs for other classes, see Figure 4.2c. Specifically, we do not zero out all connections (Section 4.1.1.2), but instead only apply the local pooling operation from DiFull and thus obtain local classification heads for each subimage (as in DiFull). However, in this setting, the classification head for a specific subimage can be influenced by features in other subimages that lie within the head's receptive field. For models with a small receptive field, this yields very similar results as DiFull (Section 4.3 and Appendix A).

[2]This is equivalent to setting the respective weights of a convolutional kernel to zero every time it overlaps with another subimage.

4.1.2 Qualitative Evaluation: AggAtt

In addition to quantitative metrics, attribution methods are often compared qualitatively on individual examples for a visual assessment. However, this is sensitive to the choice of examples and does not provide a holistic view of the method's performance. By constructing standardised grids, in which 'good' and 'bad' (GridPG) or *possible* and *impossible* (DiFull) attributions are always located in the same regions, we can instead construct *aggregate attribution maps*.

Thus, we propose a new qualitative evaluation scheme, **AggAtt**, for which we generate a set of aggregate maps for each method that progressively show the performance of the methods from the best to the worst localised attributions.

For this, we first select a grid location and then sort all corresponding attribution maps in descending order of the localisation score, see Equation (4.1). Then, we bin the maps into percentile ranges and, finally, obtain an aggregate map per bin by averaging all maps within a single bin. In our experiments, we observed that attribution methods typically performed consistently over a wide range of inputs, but showed significant deviations in the tails of the distributions (best and worst case examples). Thus, to obtain a succinct visualisation that highlights both distinct failure cases as well as the best possible results, we use bins of unequal sizes. Specifically, we use smaller bins for the top and bottom percentiles. For an example of AggAtt, see Figure 4.1.

As a result, AggAtt allows for a *systematic* qualitative evaluation and provides a holistic view of the performance of attribution methods across many samples.

4.1.3 Attributions Across Network Layers: ML-Att

Attribution methods often vary significantly in the degree to which they explain a model. Activation-based attribution methods like GradCAM [SCD+17], e.g., are typically applied on the last spatial layer, and thus only explain a fraction of the full network. This is a significantly easier task as compared to explaining the entire network, as is done by other backpropagation-based methods. Activations from deeper layers of the network would also be expected to localise better, since they would represent the detection of higher level features by the network (Figure 4.1, left). Therefore, there is a potential trade-off between the extent to which the network is explained and how well localised the attribution explanations are, which in turn would likely determine how useful the attributions are to end users.

For a *fair comparison* between methods, and to further examine this trade-off, we thus propose a multi-layer evaluation scheme for attributions (**ML-Att**). Specifically, we evaluate methods at various network layers and compare their performance *on the same layers*. For this, we evaluate all methods at the input, an intermediate, and the final spatial layer of multiple network architectures, see Section 4.2 for details. Importantly, we find that apparent differences found between some attribution methods vanish when compared *fairly*, i.e., on the same layer (Section 4.3.1).

Lastly, we note that most attribution methods have been designed to assign importance values to *input features* of the model, not *intermediate* network activations. The generalisation to intermediate layers, however, is straightforward. For this, we simply divide the full model $\mathbf{f}_{\text{full}}$ into two virtual parts: $\mathbf{f}_{\text{full}} = \mathbf{f}_{\text{explain}} \circ \mathbf{f}_{\text{pre}}$. Specifically, we treat $\mathbf{f}_{\text{pre}}$ as a pre-processing step and use the attribution methods to explain the outputs of $\mathbf{f}_{\text{explain}}$ with respect to the inputs $\mathbf{f}_{\text{pre}}(\mathbf{x})$. Note that in its standard use case, in GradCAM $\mathbf{f}_{\text{pre}}(\mathbf{x})$ is given by all convolutional layers of the model, whereas for most gradient-based methods $\mathbf{f}_{\text{pre}}(\mathbf{x})$ is the identity.

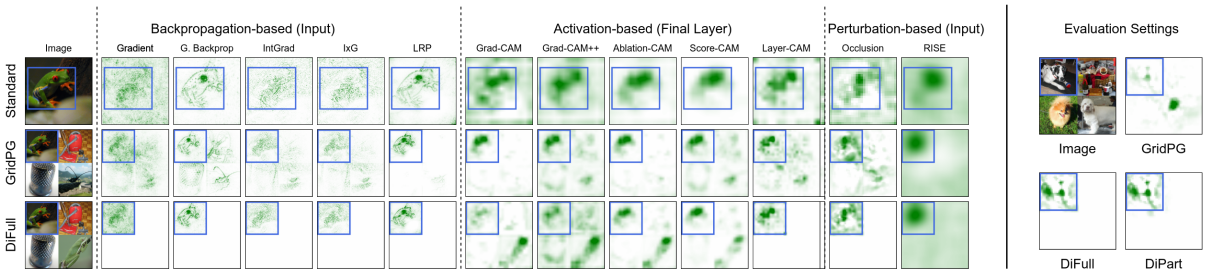


Figure 4.3: **Left:** Example Attributions on the Standard, GridPG, and DiFull Settings. We show attributions for all methods on their typically evaluated layers, i.e. input for backpropagation-based and perturbation-based, and final layer for activation-based methods. Blue boxes denote the object bounding box (Standard) or the grid cell (GridPG, DiFull) respectively. For DiFull, we use images of the same class at the top-left and bottom-right corners as in our experiments. **Right:** Occlusion attributions for an example evaluated on GridPG, DiFull, and DiPart. The top-left and bottom-right corners contain two different species of dogs, which share similar low-level features, causing both to be attributed in GridPG. In contrast, our disconnected construction in DiFull and DiPart ensures that the bottom-right subimage does not influence the classification of the top-left, and thus should not be attributed by any attribution methods, even though some do erroneously.

4.2 Experimental Setup

Dataset and Architectures. We run our experiments on VGG-19 [SZ15] and ResNet-152 [HZRS16] trained on ImageNet [DDS+09]; similar results on other architectures and on CIFAR10 [Kri09] can be found in Appendix A. For each model, we separately select images from the validation set that were classified with a confidence score of at least 0.99. By only using highly confidently classified images [BFS21, ADPGG21], we ensure that the features within each grid cell constitute positive evidence of its class for the model, and features outside it contain low positive evidence since they get confidently classified to a different class.

Evaluation on GridPG, DiFull, and DiPart. We evaluate on 2×2 grids constructed by randomly sampling images from the set of confidently classified images (see above). Specifically, we generate 2000 attributions per method for each of GridPG, DiFull, and DiPart. For GridPG, we use images from distinct classes, while for DiFull and DiPart we use distinct classes except in the bottom right corner, where we use the same class as the top left. By repeating the same class twice, we can test whether an attribution method simply highlights class-related features, irrespective of them being used by the model. Since subimages are disconnected from the classification heads of other locations in DiFull and DiPart, the use of repeating classes does not change which regions should be attributed (Section 4.1.1.2).

Evaluation at Intermediate Layers. We evaluate each method at the input, middle[3] (Conv9 for VGG-19, Conv3_x for ResNet-152), and final spatial layer (Conv16 for VGG-19, Conv5_x for ResNet-152) of each network, see Section 4.1.3. Evaluating beyond the input layer yields lower dimensional attribution maps, given by the dimensions of the respective activation maps. As is common practice [SCD+17], we thus upsample those maps to the image dimensions (448×448) using bilinear interpolation.

Qualitative Evaluation on AggAtt. As discussed, for AggAtt we use bins of unequal sizes (Section 4.1.2). In particular, we bin the attribution maps into the following percentile ranges: 0–2%, 2–5%, 5–50%,

[3] We show a single intermediate layer to visualise trends from the input to the final layer; for results on all layers, see Appendix A.

50–95%, 95–98%, and 98–100%; cf. Figure 4.1. Further, in our experiments we evaluate the attributions for classes at the top-left grid location.

Attribution Methods. We evaluate a diverse set of attribution methods, for an overview see Section 2.1 of Chapter 2. As discussed in Section 4.1.3, to apply those methods to intermediate network layers, we divide the full model into two virtual parts $\mathbf{f}_{\text{pre}}$ and $\mathbf{f}_{\text{explain}}$ and treat the output of $\mathbf{f}_{\text{pre}}$ as the input to $\mathbf{f}_{\text{explain}}$ to obtain importance attributions for those 'pre-processed' inputs. In particular, we evaluate the following methods. From the set of **backpropagation-based methods**, we evaluate on Guided Backpropagation [SDBR15], Gradient [SVZ14], IntGrad [STY17], IxG [SGK17], and LRP [BBM+15]. From the set of **activation-based methods**, we evaluate on GradCAM [SCD+17], GradCAM++ [CSHB18], AblationCAM [DR20], ScoreCAM [WWD+20], and LayerCAM [JZH+21]. Note that in our framework, these methods can be regarded as using the classification head only (except [JZH+21]) for $\mathbf{f}_{\text{explain}}$, see Section 4.1.3. To evaluate them at earlier layers, we simply expand $\mathbf{f}_{\text{explain}}$ accordingly to include more network layers. From the set of **perturbation-based methods**, we evaluate Occlusion [ZF14] and RISE [PDS18]. These are typically evaluated on the input layer, and measure output changes when perturbing (occluding) the input (Figure 4.3, left). Note that Occlusion involves sliding an occlusion kernel of size K with stride s over the input. We use $K{=}16, s{=}8$ for the input, and $K{=}5, s{=}2$ at the middle and final layers to account for the lower dimensionality of the feature maps. For RISE, we use $M{=}1000$ random masks, generated separately for evaluations at different network layers.

For LRP, following [ADPGG21, MLB+17], we primarily use a configuration that applies the ϵ-rule with $\epsilon{=}0.25$ for the fully connected layers in the network, the z^+-rule for the convolutional layers except the first convolutional layer, and the $z^{\mathcal{B}}$-rule for the first convolutional layer. We discuss the performance across other configurations, including the composite configuration proposed by [MBL+19], in Section 4.3.5. Note that since certain LRP rules, such as the z^+-rule, are not implementation invariant ([STY17]), relevance may be distributed differently for functionally equivalent models. In particular, relevance propagation through batch normalisation layers can be handled in multiple ways, such as by replacing them with 1×1 convolutions or by merging them with adjacent linear layers. In our experiments, as in [MBL+19], batch normalisation layers are handled by merging them with adjacent convolutional or fully connected layers. We further discuss some ramifications of the lack of implementation invariance to attribution localisation in Section 4.3.5 and Appendix A.

4.3 Experimental Results and Discussion

In this section, we first present the quantitative results for all attribution methods on GridPG, DiPart, and DiFull and compare their performance at multiple layers (4.3.1). Further, we present a simple smoothing mechanism that provides highly performant attributions on all three settings, and discuss architectural considerations that impact its effectiveness (4.3.3). Finally, we present qualitative results using AggAtt, and show its use in highlighting strengths and deficiencies of attribution methods (4.3.4).

4.3.1 Evaluation on GridPG, DiFull, and DiPart

We perform ML-Att evaluation using the the input (Inp), and the activations at a middle layer (Mid) and final convolutional layer (Fin) before the classification head (x-ticks in Figure 4.4) for all three quantitative evaluation settings (GridPG, DiFull, DiPart, minor columns in Figure 4.4) discussed in Section 4.1. In the following, we discuss the methods' results, grouped by their 'method family': backpropagation-based, activation-based, and perturbation-based methods (major columns in Figure 4.4).

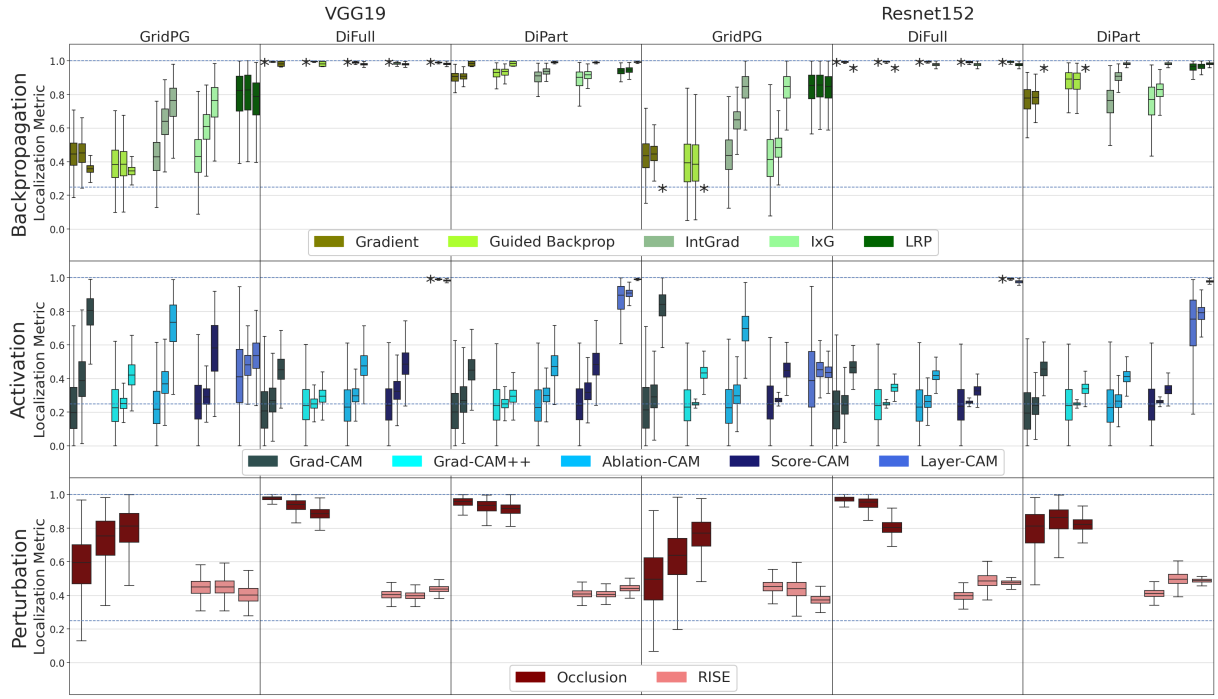


Figure 4.4: **Quantitative Results on VGG-19 and ResNet-152.** For each metric, we evaluate all attribution methods with respect to the input image (Inp), a middle (Mid), and the final (Fin) spatial layer. Boxes of the same colour correspond to the same attribution method, and each group of three boxes shows, from left to right, the results at the input (Inp), middle (Mid), and final (Fin) spatial layers respectively. We observe the performance to improve from *Inp* to *Fin* on most settings. See also Figure 4.5. We find similar results on DiFull and DiPart across methods. The symbol * denotes boxes that collapse to a single value, for better readability.

Backpropagation-based Methods. We observe that all methods except LRP perform poorly at the initial layer on GridPG (Figure 4.4, left). Specifically, we observe that they yield noisy attributions that do not seem to reflect the grid structure of the images; i.e., positive attributions are nearly as likely to be found outside of a subimage for a specific class as they are to be found inside.

However, they improve on later layers. At the final layer, IntGrad and IxG show very good localisation (comparable to GradCAM), which suggests that the methods may have similar explanatory power when compared on an equal footing. We note that IxG at the final layer has been previously proposed under the name DetGradCAM [SSPP20].

LRP, on the other hand, performs strongly at all three layers. We believe that this is likely because the z^+ rule used in the convolutional layers propagates relevance backwards in a manner that favours activations that contribute positively to the final output. As the localisation metric only considers positive attributions, such a propagation scheme would result in a high localisation score. Note that this only evaluates a single LRP configuration, as we discuss in Section 4.3.5, we find that the performance can significantly vary based on the propagation rules used.

On DiFull, all backpropagation-based methods show near-perfect localisation across layers. No attribution is given to disconnected subimages since the gradients with respect to them are zero (after all, they are *fully disconnected*); degradations for other layers can be attributed to the applied upsampling. However, the lack of implementation invariance [STY17] in LRP implies that relevance could be made to effectively propagate through disconnected regions by constructing an appropriate functionally equivalent model, as

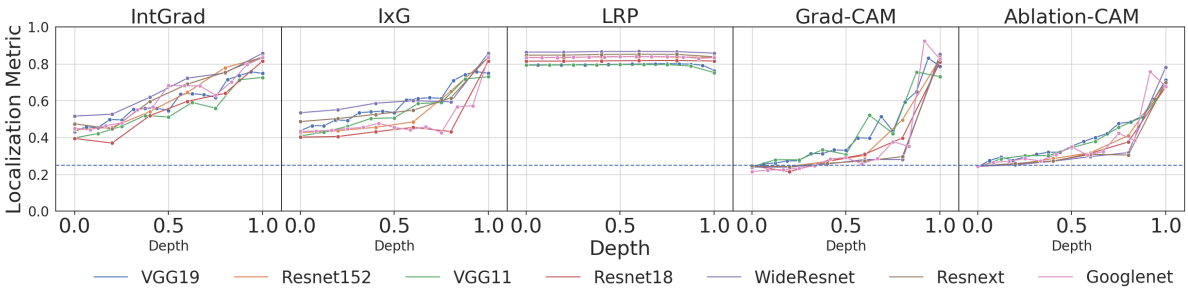


Figure 4.5: **Mean localisation performance layer-wise across seven models of a selected subset of attribution methods.** For each method and network, we plot the mean localisation score at at several depths. The x-axis shows the fraction of the total network depth (0 - input, 1 - final layer). As discussed in Section 4.1.3 and Figure 4.4, the localisation performance tends to improve towards the final layer.

we discuss in Section 4.3.5 and Appendix A.

Similar results are seen in DiPart, but with decreasing localisation when moving backwards from the classifier, which can be attributed to the fact that the receptive field can overlap with other subimages in this setting. Overall, we find that similar performance is obtained on DiFull and DiPart across all methods.

Activation-based Methods. We see that all methods with the exception of LayerCAM improve in localisation performance from input to final layer on all three settings. Since attributions are computed using a scalar weighted sum of attribution maps, this improvement could be explained by improved localisation of activations from later layers. In particular, localisation is very poor at early layers, which is a well-known limitation of GradCAM [JZH+21]. The weighting scheme also causes final layer attributions for all methods except LayerCAM to perform worse on DiFull than on GridPG, since these methods attribute importance to both instances of the repeated class (cf. Figure 4.9). This issue is absent in LayerCAM as it does not apply a pooling operation.

Perturbation-based Methods. We observe (Figure 4.4, right) Occlusion to perform well across layers on DiFull, since occluding disconnected subimages cannot affect the model outputs and are thus not attributed importance. However, the localisation drops slightly for later layers. This is due to the fact that the relative size (w.r.t. the activation map) of the overlap regions between occlusion kernels and adjacent subimages increases. This highlights the sensitivity of performance to the choice of hyperparameters, and the tradeoff between computational cost and performance.

On GridPG, Occlusion performance improves with layers. On the other hand, RISE performs poorly across all settings and layers. Since it uses random masks, pixels outside a target grid cell that share a mask with pixels within get attributed equally. So while its attributions concentrate more in the target grid cell, a significant amount of positive attributions can be found outside of it, see also Figure 4.9.

4.3.2 Localisation Across Network Depths

In this section, we evaluate the trends in localisation performance across the full range of network depths for the seven models we evaluate on (VGG-19, VGG-11 [SZ15], ResNet-152, ResNet-18 [HZRS16], ResNeXt [XGD+17], Wide ResNet [ZK16], GoogLeNet [SLJ+15]). Our quantitative evaluation using our proposed ML-Att scheme so far (Figure 4.4) focused on three representative network depths – at the input, a middle layer, and the final layer of each model. We found that several methods (e.g. IxG, IntGrad, GradCAM, LRP) localise well at the final layer. Here, we evaluate whether the performance on these three

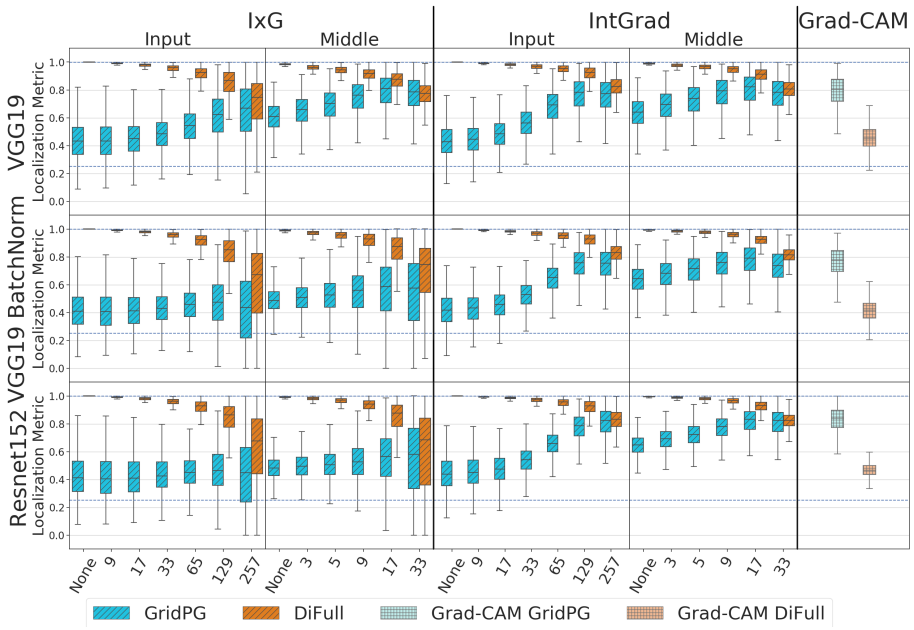


Figure 4.6: **Smoothing the attributions** for IntGrad and IxG significantly improves their performance at the input image and middle layer. For reference, we show GradCAM on the *final* spatial layer.

layers is representative of the general trend across all layers, and whether the trends for each attribution methods generalise across diverse network architectures.

The quantitative results for a subset of attribution methods can be found in Figure 4.5; for the remaining methods, see Appendix A. We pick four methods, two backpropagation-based (IntGrad, IxG) and two activation-based (GradCAM, AblationCAM), whose performance increases most prominently from the input to the final layer in Figure 4.4. In addition, we show results on LRP, the best performing method overall. Full results on all methods can be found in Appendix A. For each attribution method, we plot the mean localisation score on each model across all network depths. The x-axis shows the fraction of the model depth, where 0 refers to the input layer and 1 refers to the final convolutional layer, and the y-axis shows the localisation score. Each line plots the mean localisation score across all possible depths for a single model.

We find that the trends in performance at the chosen three layers in Figure 4.4 generalise to all layers, with the localisation performance improving at deeper layers for all the chosen methods (except LRP). Furthermore, we find that these trends also generalise across network architectures, and demonstrates the utility of ML-Att in finding similar performance across diverse attribution methods when compared fairly at identical depths. We find that the performance of IntGrad and IxG steadily improves from the input to the final layer, while that of GradCAM and AblationCAM is poor except near the final layer. LRP, on the other hand, scores highly throughout the network.

4.3.3 Smoothing Attributions

From Section 4.3.1, we see that GradCAM localises well at the final layer in GridPG, but performs poorly on all the other settings as a consequence of global pooling of gradients (for DiFull) and poor localisation of early layer features (for GridPG early layers). Since IxG, in contrast, does not use a pooling operation, it performs well on DiFull at all layers and on GridPG at the final layer. However, it performs poorly at the input and middle layers on GridPG due to the noisiness of gradients; IntGrad shows similar results.

Devising an approach to eliminate this noise would provide an attribution method that performs well across settings and layers. Previous approaches to reduce noise include averaging attribution maps over many perturbed samples (SmoothGrad [STK+17], see Appendix A for a comparison) or adding a gradient penalty during training [KTI19]. However, SmoothGrad is computationally expensive as it requires several passes on the network to obtain attributions, and is sensitive to the chosen perturbations. Similarly, adding a penalty term during training requires retraining the network.

Here, we propose to simply apply a Gaussian smoothing kernel on existing IntGrad and IxG attributions. We evaluate on DiFull and GridPG using several kernel sizes, using standard deviation $K/4$ for kernels of size K. We refer to the smooth versions as S-IntGrad and S-IxG respectively.

On VGG-19 (Figure 4.6, top), we find that S-IntGrad and S-IxG localise significantly better than IntGrad and IxG, and the performance improves with increasing kernel size. In detail, S-IntGrad *on the input layer* with $K = 257$ outperforms GradCAM *on the final layer*, despite explaining the full network. While performance on DiFull drops slightly as smoothing leaks attributions across grid boundaries, both S-IntGrad and S-IxG localise well across settings and layers. However, on ResNet-18 (Figure 4.6, bottom), while S-IntGrad improves similarly, S-IxG does not, which we discuss next.

Impact of Network Architecture. A key difference between the VGG-19 and ResNet-152 architectures used in our experiments is that VGG-19 does not have batch normalisation (BatchNorm) layers. We note that batch norm effectively randomises the sign of the input vectors to the subsequent layer, by centring those inputs around the origin (cf. [IS15, KTI19]). Since the sign of the input determines whether a contribution (weighted input) is *positive* or *negative*, a BatchNorm layer will randomise the sign of the contribution and the 'valence' of the contributions will be encoded in the BatchNorm biases. To test our hypothesis, we evaluate S-IxG on a VGG-19 with BatchNorm layers (Figure 4.6, middle), and observe results similar to ResNet-152: i.e., we observe no systematic improvement by increasing the kernel size of the Gaussian smoothing operation. This shows that the architectural choices of a model can have a significant impact on the performance of attribution methods.

4.3.4 Qualitative Evaluation Using AggAtt

In this section, we present qualitative results using AggAtt for select attributions evaluated on GridPG and DiFull and multiple layers. First, to investigate the qualitative impact of smoothing, we use AggAtt to compare IxG, S-IxG, and GradCAM attributions on GridPG on multiple layers. We employ AggAtt on DiFull to highlight specific characteristics and failure cases of some attribution methods.

AggAtt on GridPG. We show AggAtt results for IxG, S-IxG, GradCAM, and LRP at three layers on GridPG using VGG-19 on the images at the top-left corner (Figure 4.7). For each method, a set of three rows corresponds to the attributions at input, middle, and final layers. For S-IxG, we set K to 129, 17, and 9 respectively. We further show individual samples (median bin) of the first and last bins per method.

We observe that the aggregate visualisations are consistent with the quantitative results (Figures 4.4 and 4.6) and the individual examples shown for each bin. The performance improves for IxG and GradCAM from input to final layer, while S-IxG localises well across three layers. Attributions from LRP are generally visually pleasing and localise well across layers. Finally, the last two columns show that all the attribution methods perform 'poorly' for some inputs; e.g., we find that IxG and GradCAM on the final layer attribute importance to other subimages if they exhibit features that are consistent with the class in the top-left subimage. While the attributions might be conceived as incorrect, we find that many 'failure cases' on GridPG highlight features that the underlying model might in fact use, even if they are in another subimage. Given the lack of ground truth, it is difficult to assess whether these attributions

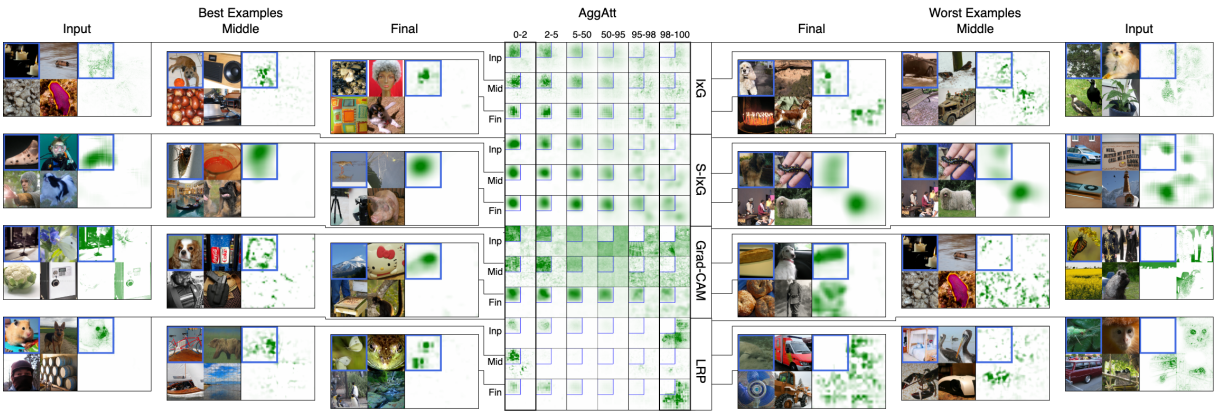


Figure 4.7: **Qualitative Results for VGG-19 on GridPG evaluated at the top-left corner**. *Centre:* Aggregate attributions sorted and binned in descending order of localisation. Each column corresponds to a bin, and set of three rows corresponds to a method. For each method, the three rows from top to bottom show the aggregate attributions at the input, middle, and final spatial layers. *Left:* Examples from the first bin, which corresponds to the best set of attributions. *Right:* Similarly, we show examples from the last bin, which corresponds to the worst set of attributions. For smooth IxG, we use $K = 129$ for the input layer, $K = 17$ at the middle layer, and $K = 9$ at the final layer. All examples shown correspond to images whose attributions lie at the median position in their respective bins.

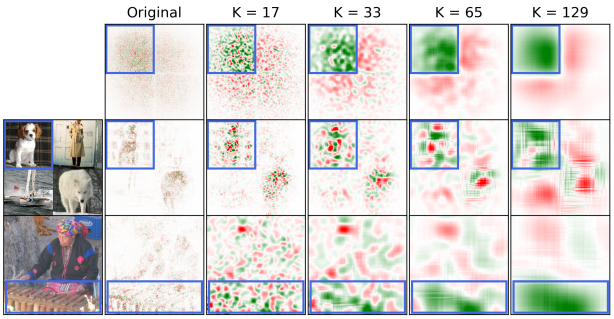


Figure 4.8: **Qualitative Visualisation of S-IxG attributions for various kernel sizes, including both positive and negative attributions.** *Top*: Aggregate attribution maps for VGG-19 on GridPG at the top-left corner across the dataset. We see that positive attributions (green) aggregate to the top-left grid cell and negative attributions (red) aggregate outside when smoothing with large kernel sizes. *Middle and Bottom*: Examples of smoothing on a single grid and non-grid image. Positive attributions concentrate inside the bounding box when smoothed with large kernels.

faithfully reflect model behaviour or deficiencies of the attribution methods.

Despite explaining significantly more layers, S-IntGrad and S-IxG at the input layer not only match GradCAM at the final layer quantitatively (Figure 4.6) and qualitatively (Figure 4.7), but are also highly consistent with it for individual explanations. Specifically, the Spearman rank correlation between the localisation scores of GradCAM (final layer) and S-IntGrad (input layer) increases significantly as compared to IntGrad (input layer) (e.g., $0.3 \rightarrow 0.78$ on VGG-19), implying that their attributions for any input tend to lie in the same AggAtt bins (see Appendix A).

To further understand the effect of smoothing, we visualise S-IxG with varying kernel sizes while including negative attributions (Figure 4.8). The top row shows aggregate attributions across the dataset, while the middle and bottom rows show an example under the GridPG and standard localisation settings respectively. We observe that while IxG attributions appear noisy (column 2), smoothing causes positive and negative attributions to cleanly separate out, with the positive attributions concentrating around the object. For instance, in the second row, IxG attributions concentrate around both the dog and the wolf, but S-IxG with $K = 129$ correctly attributes only the dog positively. This could indicate a limited effective receptive field (RF) [LLUZ16] of the models. Specifically, note that for piece-wise linear models, summing the

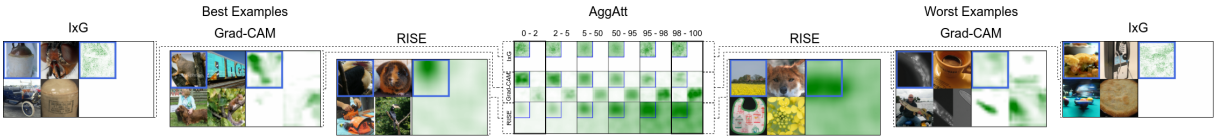


Figure 4.9: **Qualitative Results for VGG-19 on DiFull evaluated at the top-left corner**. *Centre:* Aggregate attributions sorted and binned in descending order of localisation. Each column corresponds to a bin and each row corresponds to a method applied at its standard layer. *Left:* Examples from the first bin, which corresponds to the best set of attributions. *Right:* Examples from the last bin, which corresponds to the worst set of attributions. All examples shown correspond to images whose attributions lie at the median position in their bins.

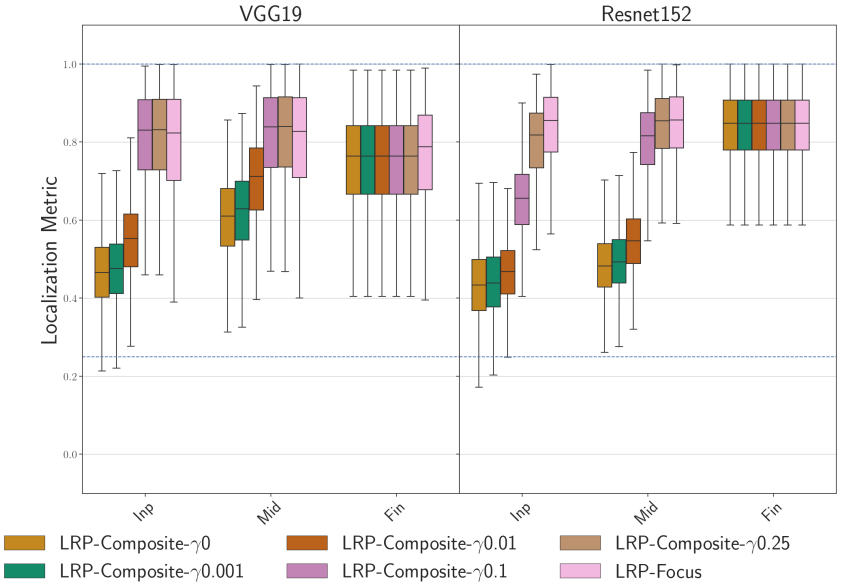


Figure 4.10: **Quantitative Results for various LRP configurations on VGG-19 and ResNet-152.** For each metric, we evaluate the attributions with respect to the input (Inp), a middle (Mid), and the final (Fin) spatial layer.

contributions (given by IxG) over all input dimensions within the RF exactly yields the output logit (disregarding biases). Models with a small RF would thus be well summarised by S-IxG for an adequately sized kernel; we elaborate on this in Appendix A.

AggAtt on DiFull. We visually evaluate attributions on DiFull for one method per method family, i.e., from backpropagation-based (IxG, input layer), activation-based (GradCAM, final layer), and perturbation-based (RISE, input layer) methods at their standard layers (Figure 4.9). The top row corroborates the near-perfect localisation shown by the backpropagation-based methods on DiFull. The middle row shows that GradCAM attributions concentrate at the top-left and bottom-right corners, which contain images of the same class, since global pooling of gradients makes it unable to distinguish between the two even though only the top-left instance (here) influences classification. Finally, for RISE, we observe that while attributions localise well for around half the images, the use of random masks results in noisy attributions for the bottom half.

4.3.5 Evaluation Using Various LRP Configurations

From the previous sections, we saw that LRP using the configuration by [ADPGG21] outperformed all

other attribution methods at all layers. More generally, LRP [BBM+15] is a paradigm that encompasses a family of attribution methods that modify the gradients during backpropagation. The mechanism of relevance propagation is specified by a set of propagation rules used across the network. Rules are selected for each layer usually based on the type of layer and its position in the network, and a mapping of layers to rules constitutes a unique LRP configuration. Some of the existing backpropagation-based methods that were proposed independently, such as IxG [SGK17] and Excitation Backprop [ZBL+18], can be viewed as specific configurations of LRP [MBL+19].

In this section, we study the impact of the choice of rules and their hyperparameters in attribution performance of LRP. Specifically, following prior work [MBL+19], we consider a composite configuration (hereafter referred to as LRP-Composite), that applies the ϵ-rule on fully connected layers, the γ-rule on convolutional layers except the first layer, and the $z^{\mathcal{B}}$-rule on the first convolutional layer. In contrast to the ϵ-rule that weighs positive and negative contributions equally when propagating relevance, the γ-rule uses a hyperparameter γ that increases the weight given to positive contributions. As $\gamma \to \infty$, relevance is propagated only based on positive contributions, and the configuration is identical to the one used in [ADPGG21] and the previous sections (hereafter referred to as LRP-Focus). In our experiments, we investigate the impact of γ on performance of LRP, and evaluate LRP-Composite using values of γ in $\{0, 0.001, 0.01, 0.1, 0.25\}$. $\gamma = 0$ corresponds to using the ϵ-rule where no additional weight is given to positive contributions, and $\gamma = 0.25$ is the value that is commonly used (e.g. [MBL+19]). We also evaluate the setting when $\gamma \to \infty$, i.e. using LRP-Focus. Quantitative results for both models on GridPG can be found in Figure 4.10.

We find that the performance is highly sensitive to the choice of γ. Low values of γ (up to 0.01) localise poorly, particularly at the input layer. For higher values of γ, including LRP-Focus where $\gamma \to \infty$, the localisation performance is high across layers for both models on GridPG. We attribute this to the following: if only positive contributions are considered at intermediate layers, the *sign* of the attributions to the last layers will be maintained throughout the backpropagation process. In particular, the distribution of positive and negative attributions at the input layer will be largely dependent on the attributions at the final layer. Hence, since the ϵ-rule performs well at the final layer (similar to IxG and IntGrad), maintaining the sign of the attributions will lead to good results at the input layer, which the γ-rule achieves by suppressing negative contributions. We believe that understanding how to better integrate the negative contributions in the backward pass to reflect all model computations is thus an interesting direction to explore in future work.

Lack of Implementation Invariance. As discussed in [STY17], LRP in general is not implementation invariant, i.e., functionally equivalent models could be assigned highly dissimilar attribution maps for the same input. In particular, this also holds for the z^+-rule, which is used in the best-performing LRP-Focus configuration. This leads to the possibility of controlling which pixels get attributed by appropriately formulating an equivalent model. Importantly, as we show in Appendix A, pixels that have no influence on the output can thus also get high attributions. This shows that while LRP can be highly performant, one must carefully consider the parameters used and the properties of the setting when using it in practice.

4.4 CONCLUSION

In this section, we summarise our results, and discuss high-level recommendations. First, we proposed a novel quantitative evaluation setting, DiFull, to disentangle the behaviour of the model from that of the attribution method. This allowed us to evaluate for model-faithfulness by partitioning inputs into regions that could and could not influence the model's decision. Using this, we showed that (Figure 4.4) some popularly used attribution methods, such as GradCAM, can provide model-unfaithful attributions. On the

other hand, while noisy, backpropagation-based methods like IntGrad and IxG localise perfectly under this setting. We note, however, that our setting cannot evaluate the correctness of attributions within the target grid cells, and as such a high localisation performance on DiFull is a necessary condition for a good attribution method, but not a sufficient condition. In other words, DiFull can be viewed as a coarse sanity check that should be passed by any model-faithful attribution method, but our results show that several do not do so. This could be of practical importance in use cases where models learn to focus on a fixed local region in an image to reach their decisions.

Second, we observed that different attribution methods are typically evaluated at different depths, which leads to them being compared unfairly. To address this, we proposed a multi-layer evaluation scheme, ML-Att, through which we compared each attribution method at identical model depths (Figures 4.4 and 4.5). We found that surprisingly, a diverse set of methods perform very similarly and localise well, particularly at the final layer. This includes backpropagation-based methods like IxG and IntGrad, which have often been criticised for providing highly noisy and hard to interpret attributions. Combined with their perfect localisation on DiFull, this shows that IxG and IntGrad at the final layer can be used as an alternative to GradCAM, when coarse localisation is desired. Quantitative (Figures 4.4 and 4.5) and qualitative (Figures 4.7 and 4.9) results at intermediate layers also point to the existence of a trade-off between faithfulness and coarseness of attributions, particularly for methods like IxG and IntGrad. While attributions computed closer to the input explain a larger fraction of the network and provides more fine-grained attributions, such attributions often localise poorly and are not very helpful to end users. On the other hand, attributions computed closer to the final layer explain only a small part of the network, but are coarser, localise better and highlight the object features more clearly. As a result, the choice of layer to compute attributions would depend on the user's preference in the presence of this trade-off.

Third, we proposed an aggregate attribution evaluation scheme, AggAtt, to holistically visualise the performance of an attribution method. Unlike evaluation on a small subset of examples, this shows the full range of localisations across the dataset and eliminates any inadvertent biases from the choice of examples. Furthermore, it allows one to easily visualise the performance at the best and worst localised examples, and could help identify cases when an attribution method unexpectedly fails.

Fourth, we showed that a simple post-hoc Gaussian smoothing step can significantly improve localisation (Figures 4.6 and 4.8) for some attribution methods (IntGrad, IxG). Unlike commonly used smoothing techniques like SmoothGrad, this requires no additional passes through the network and no selection of hyperparameters. As we show in Appendix A, it also results in better localised attributions. This shows that while originally noisy, obtaining a local summary of attribution maps from these methods could provide maps that are useful for humans in practice. However, we find that the effectiveness of smoothing is influenced by the network architecture, in particular the presence of batch normalisation layers, which suggests that architectural considerations must be taken into account when using attribution methods.

Finally, we find that certain configurations of layer-wise relevance propagation (LRP) consistently perform the best quantitatively and qualitatively across network depths. However, by interpolating between different LRP configurations (see Section 4.3.5), we find that this is likely due to the fact that the well-performing LRP-configurations maintain the sign of the attributions to the final layer in the backpropagation process. As such, some aspects of the model computations are not reflected in the final attribution maps (negative contributions at intermediate layers are neglected) and the final attributions are largely dependent on the localisation performance at the final layer. How to better reflect those negative contributions in the backpropagation process is thus an interesting direction for future work.

While we focus on CNNs in our work, performing a comprehensive evaluation for attribution methods on the recently proposed state-of-the-art image classification architectures such as vision transformers (ViTs) [DBK+21] is another interesting direction for future work.

Overall, we find that fair comparisons, holistic evaluations (DiFull, GridPG, AggAtt, ML-Att), and careful disentanglement of model behaviour from the explanations provide better insights in the performance of attribution methods.

II

Designing Inherently Interpretable Deep Neural Networks

In the preceding chapters, we found that while recent *post-hoc* attribution methods can be a powerful tool for better understanding DNN decisions (Chapter 3), it also became clear that it remains a challenge to faithfully interpret the full DNN models in a reliable manner (Chapter 4). We believe this is, to some degree, inherent to the post-hoc approach itself: the DNNs under consideration have simply not been designed and optimised to be easily interpretable. Moreover, by definition, post-hoc approaches do not take the model optimisation and task formulation into account, despite the fact that these, of course, critically determine the optima that the models converge to.

Therefore, in the following two chapters, we explore a different path towards better understanding the predictions of DNNs. In particular, instead of trying to interpret DNNs after they were trained, we explicitly design and optimise novel DNN architectures such that they can inherently be interpreted in a faithful and easily understandable manner. Importantly, this puts the goal of model interpretability first and fundamentally changes how the DNNs operate. As a result, our proposed models do not need to be interpreted *post-hoc*, but instead inherently compute their outputs in an easily interpretable manner.

Specifically, in **Chapter 5**, we present the CoDA Networks, which introduce the core concepts on which the interpretability of our proposed models relies: dynamic linearity and alignment pressure. Further, in **Chapter 6** we distill these properties into a single operation—named the B-cos transformation—that can be used as a drop-in replacement for the ubiquitously used linear transformation and thus allows us to significantly increase the interpretability of a wide range of state-of-the-art DNN architectures. Importantly, as we show in Chapter 6, this is possible at only minor costs in classification performance.

5 Convolutional Dynamic Alignment Networks

Contents

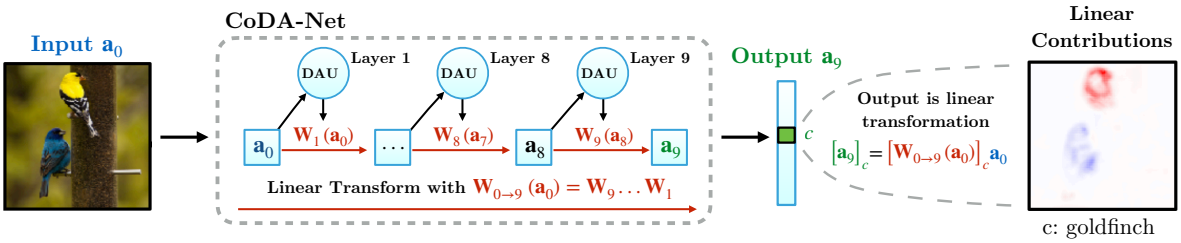


Figure 5.1: Sketch of a 9-layer CoDA Net, which computes its **output** $\mathbf{a}_9$ for an **input** $\mathbf{a}_0$ as a linear transform via a matrix $\mathbf{W}_{0\to 9}(\mathbf{a}_0)$. As such, the output can be linearly decomposed into input contributions (see right). This global transformation matrix $\mathbf{W}_{0\to 9}$ is computed successively via multiple layers of Dynamic Alignment Units (DAUs), cf. Equation (5.10). These layers, in turn, produce intermediate linear transformation matrices $\mathbf{W}_l(\mathbf{a}_{l-1})$ that align with the inputs of layer l. As a result, the combined matrix $\mathbf{W}_{0\to 9}$ also aligns with task-relevant patterns. Positive (negative) contributions for the class 'goldfinch' are shown in red (blue). These contribution maps are highly class-specific: e.g., see Figure 5.2 for the map corresponding to the indigo bunting class (blue bird).

As discussed in the previous chapters, explaining the predictions of Deep Neural Networks (DNNs) in a model-faithful manner is an open research problem. Linear models, on the other hand, are generally considered inherently interpretable, because the *contribution* ('the weighted input') of every dimension to the output is explicitly given. Interestingly, many modern DNNs implicitly model the output as a linear transformation of the input; a ReLU-based [NH10] neural network, e.g., is piece-wise linear and the output thus a linear transformation of the input, cf. [MPCB14]. However, due to the highly non-linear manner in which these linear transformations are 'chosen', the corresponding contributions per input dimension are very noisy and not easily understandable for humans, see for example Figure 5.2.

In this chapter, we introduce a novel network architecture, the **Convolutional Dynamic Alignment Networks (CoDA Nets)**, for which the model-inherent contribution maps are not only faithful projections of the internal computations, but also easily understandable for humans and thus good explanations of the model prediction. There are two main components to the interpretability of the CoDA Nets. First, the CoDA Nets are dynamic linear, i.e., they compute their outputs through a series of input-dependent linear transforms, which are based on our novel Dynamic Alignment Units (DAUs). As in linear models, the output can thus be decomposed into individual input contributions, see Figure 5.1. Second, the DAUs are structurally biased to compute weight vectors that align with relevant patterns in their inputs. In combination, the CoDA Nets thus inherently produce contribution maps that are 'optimised for interpretability': since each linear transformation matrix is optimised to align with discriminative features, the contribution maps reflect the most discriminative features *as used by the model*. The contribution maps of CoDA Nets thus lend themselves better to be used as easily understandable explanations for humans than those of piece-wise linear models, which generally do not align well with discriminative input patterns (cf. Figure 5.2), despite accurately summarising the models.

In short, we present a new direction for building inherently more interpretable neural network architectures with high modelling capacity. In detail, we would like to highlight the following contributions:

(1) We introduce the concept of Dynamic Alignment Units (DAUs), which improve the interpretability of neural networks. They have two key properties: they are *dynamic linear* and align their dynamically computed weights with their inputs.

(2) Specifically, this weight alignment is induced by constraining the norm of the dynamically computed weights. We show that this can either be done explicitly by fixing the dynamic weights to be of unit norm, or by constraining an *upper bound* of their norm, which results in more efficient DAUs.

(3) We introduce Convolutional Dynamic Alignment Networks (CoDA Nets), which are built out of multiple layers of DAUs and show that the resulting networks *inherit* the dynamic linearity and the alignment properties from their constituent DAUs. As a result, the dynamically computed weights of CoDA Nets align with discriminative patterns in the input.

(4) We show that the resulting contribution maps perform well under commonly employed *quantitative* criteria for attribution methods. Moreover, under *qualitative* inspection, we note that they exhibit a high degree of detail.

(5) We analyse how different DAU formulations affect the models in terms of accuracy, interpretability, and efficiency.

(6) We show that CoDA Nets are performant classifiers and yield competitive classification accuracies on the CIFAR10 and TinyImageNet datasets.

(7) We show that CoDA Nets can be seamlessly combined with conventional networks. The resulting hybrid networks exhibit an increased 'interpretable depth' whilst taking advantage of the efficiency and strong modelling capacity of the base networks.

This chapter is based on [BFS21, BFS22a] and the corresponding code is publicly available at: github.com/moboehle/CoDA-Nets.

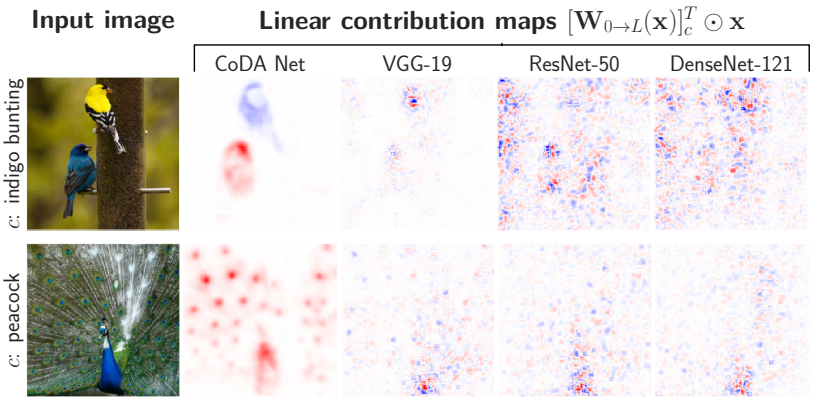


Figure 5.2: **Contribution maps of CoDA Nets and various piece-wise linear models** (VGG-19 [SZ15], ResNet-50 [HZRS16], DenseNet-121 [HLVDMW17]). While piece-wise linear models are also *dynamic linear*, their contribution maps (given by 'Input×Gradient', cf. [AGM+18]) are not as easily interpretable. E.g., in contrast to the noisy contribution maps of piece-wise linear models (*cols. 3-5*), CoDA Nets clearly separate positive and negative 'evidence' in their contribution maps (e.g., *1st row, 2nd column*). Moreover, the contribution maps of CoDA Nets are highly class-specific and depend on which class is explained (compare first row to Figure 5.1). Finally, given their high level of detail, the CoDA Nets can highlight the importance of fine-grained features, such as the eyes in the feathers of the peacock (*2nd row*). Positive (negative) contributions are shown in red (blue) for all contribution maps, and the maps have been slightly smoothed for better visibility.

5.1 DYNAMIC ALIGNMENT NETWORKS

In this section, we present our novel type of network architecture: the Convolutional Dynamic Alignment Networks (CoDA Nets). For this, we first introduce Dynamic Alignment Units (DAUs) as the basic building blocks of CoDA Nets and discuss two of their key properties in Section 5.1.1. Concretely, we show that these units linearly transform their inputs with dynamic (input-dependent) weight vectors and, additionally, that they are biased to align these weights with the input during optimisation. Given the computational costs of evaluating DAUs, in Section 5.1.2 we further present an alternative formulation of the DAUs for increased efficiency. We then discuss how DAUs can be used for classification (Section 5.1.3) and how we build performant networks out of multiple DAU layers (Section 5.1.4). Importantly, the resulting *linear decompositions* of the network outputs are optimised to align with discriminative patterns in the input, making them highly suitable for interpreting the network predictions.

We structure this section around the following **three important properties** (P1-P3) of the DAUs:

P1: Dynamic linearity. The DAU output o is computed as a dynamic (input-dependent) linear transformation of the input $\mathbf{x}$, such that $o = \mathbf{w}(\mathbf{x})^T\mathbf{x} = \sum_j w_j(\mathbf{x})x_j$. Hence, o can be decomposed into contributions from individual input dimensions, which are given by $w_j(\mathbf{x})x_j$ for dimension j.

P2: Alignment maximisation. Maximising the average output of a single DAU over a set of inputs $\mathbf{x}_i$ maximises the alignment between inputs $\mathbf{x}_i$ and the weight vectors $\mathbf{w}(\mathbf{x}_i)$. As the modelling capacity of $\mathbf{w}(\mathbf{x})$ is restricted, $\mathbf{w}(\mathbf{x})$ will encode the most frequent patterns in the set of inputs $\mathbf{x}_i$.

P3: Inheritance. When combining multiple DAU layers to form a Dynamic Alignment Network (DA Net), the properties P1 and P2 are *inherited*: DA Nets are dynamic linear (P1) and maximising the last layer's output induces an output maximisation in the constituent DAUs (P2).

These properties increase the interpretability of a DA Net, such as a CoDA Net (Section 5.1.4) for the following reasons. First, the output of a DA Net can be decomposed into contributions from the individual input dimensions, similar to linear models (cf. Figure 5.1, P1 and P3). Second, we note that optimising a

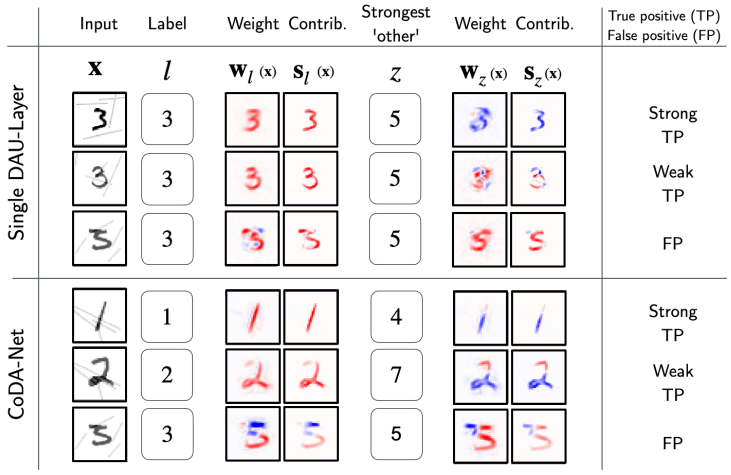


Figure 5.3: **Linear weights and contributions for different $\mathbf{x}$** (for the single layer, see Equation (5.7), for the CoDA Net Equation (5.11)) for the ground truth label l and the strongest non-label output z. As can be seen, the weights align well with the inputs. The first three rows are based on a single DAU layer, the last three on a 5 layer CoDA Net. The first two samples (rows) per model are correctly classified and the last one is misclassified.

neural network for classification applies a maximisation to the outputs of the last layer for every sample. This maximisation aligns the dynamic weight vectors $\mathbf{w}(\mathbf{x})$ of the constituent DAUs of the DA Net with their respective inputs (cf. Figure 5.3 as well as **P2** and **P3**).

Importantly, the weight vectors will align with the *discriminative* patterns in their inputs when optimised for classification as we show in Section 5.1.3. As a result, the model-inherent contribution maps of CoDA Nets are optimised to align well with *discriminative input patterns* in the input image and the interpretability of our models thus forms part of the global optimisation procedure.

5.1.1 Dynamic Alignment Units

We define the Dynamic Alignment Units (DAUs) by

$$\mathrm{DAU}(\mathbf{x}) = g(\mathbf{A}\mathbf{B}\mathbf{x} + \mathbf{b})^T\mathbf{x} = \mathbf{w}(\mathbf{x})^T\mathbf{x} \quad . \tag{5.1}$$

Here, $\mathbf{x} \in \mathbb{R}^d$ is an input vector, $\mathbf{A} \in \mathbb{R}^{d\times r}$ and $\mathbf{B} \in \mathbb{R}^{r\times d}$ are trainable transformation matrices, $\mathbf{b} \in \mathbb{R}^d$ a trainable bias vector, and $g(\mathbf{u}) = \alpha(\|\mathbf{u}\|)\mathbf{u}$ is a non-linear function that scales the norm of its input. In contrast to using a single matrix $\mathbf{M} \in \mathbb{R}^{d\times d}$, using $\mathbf{AB}$ allows us to control the maximum rank r of the transformation and to reduce the number of parameters; we will hence refer to r as the rank of a DAU. As can be seen by the right-hand side of Equation (5.1), the DAU linearly transforms the input $\mathbf{x}$ (**P1**). At the same time, given the quadratic form ($\mathbf{x}^T\mathbf{B}^T\mathbf{A}^T\mathbf{x}$) and the rescaling function $\alpha(\|\mathbf{u}\|)$, the output of the DAU is a non-linear function of its input. In the context of DAUs, we are particularly interested in functions that constrain the norm of the weight vectors $\mathbf{w}(\mathbf{x})$, such as, e.g., rescaling to unit norm (L2) or the squashing function (SQ, see [SFH17]):

$$\mathrm{L2}(\mathbf{u}) = \frac{\mathbf{u}}{\|\mathbf{u}\|_2} \quad \text{and} \quad \mathrm{SQ}(\mathbf{u}) = \mathrm{L2}(\mathbf{u}) \times \frac{\|\mathbf{u}\|_2^2}{1 + \|\mathbf{u}\|_2^2} \tag{5.2}$$

In Section 5.1.2, we further present an approximation to these rescaling functions, which lowers the computational cost of the DAUs whilst maintaining their bounding property $\|\mathbf{w}(\mathbf{x})\| \leq 1$. Given such a bound on $\mathbf{w}(\mathbf{x})$, the output of the DAUs will be upper-bounded by the norm of the input:

$$\mathrm{DAU}(\mathbf{x}) = \|\mathbf{w}(\mathbf{x})\|\,\|\mathbf{x}\|\cos(\angle(\mathbf{x}, \mathbf{w}(\mathbf{x}))) \leq \|\mathbf{x}\| \tag{5.3}$$

As a corollary, for a given input $\mathbf{x}_i$, the DAUs can only achieve this upper bound if $\mathbf{x}_i$ is an eigenvector (EV) of the linear transform $\mathbf{ABx} + \mathbf{b}$. Otherwise, the cosine in Equation (5.3) will not be maximal[1]. As can

[1]Note that $\mathbf{w}(\mathbf{x})$ is proportional to $\mathbf{ABx} + \mathbf{b}$. The cosine in Equation (5.3), in turn, is maximal if and only if $\mathbf{w}(\mathbf{x}_i)$ is proportional to $\mathbf{x}_i$ and thus, by transitivity, if $\mathbf{x}_i$ is proportional to $\mathbf{ABx}_i + \mathbf{b}$. This means that $\mathbf{x}_i$ has to be an EV of $\mathbf{ABx} + \mathbf{b}$

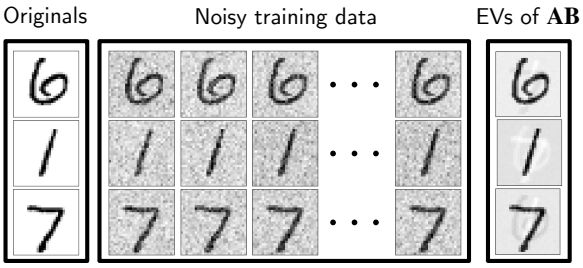


Figure 5.4: **Eigenvectors (EVs) of $\mathbf{AB}$** after maximising the output of a rank-3 DAU over a set of noisy samples of 3 MNIST digits. Effectively, the DAUs encode the most frequent components in their EVs, similar to a principal component analysis (PCA).

be seen in Equation (5.3), maximising the average output of a DAU over a set of inputs $\{\mathbf{x}_i \mid i = 1, ..., n\}$ maximises the alignment between $\mathbf{w}(\mathbf{x})$ and $\mathbf{x}$ (P2). In particular, it optimises the parameters of the DAU such that the *most frequent input patterns* are encoded as EVs in the linear transform $\mathbf{ABx} + \mathbf{b}$, similar to an r-dimensional PCA decomposition (r the rank of $\mathbf{AB}$). As an illustration of this property, in Figure 5.4 we show the 3 EVs[2] of matrix $\mathbf{AB}$ (with rank $r=3$, bias $\mathbf{b}=\mathbf{0}$) after optimising a DAU over a set of n noisy samples of 3 specific MNIST [LCB10] images; for this, we used $n=3072$ and zero-mean Gaussian noise. As expected, the EVs of $\mathbf{AB}$ encode the original, noise-free images, since this on average maximises the alignment (Equation (5.3)) between the weights $\mathbf{w}(\mathbf{x}_i)$ and the inputs $\mathbf{x}_i$ over the dataset.

5.1.2 Efficient DAUs: Bounding the Bound

As discussed in the previous section, we introduce a norm constraint for the DAU weights $\mathbf{w}(\mathbf{x})$ to ensure that large outputs can only be achieved for well-aligned weights. However, the explicit norm constraint on $\mathbf{w}(\mathbf{x})$ requires its explicit calculation, which we have observed to significantly impact the evaluation time of DAUs. Therefore, we evaluate an additional formulation of the DAUs in which we only *constrain an upper bound* of the norm of $\mathbf{w}(\mathbf{x})$. For this, we take advantage of the following inequality:

$$\|\mathbf{w}(\mathbf{x})\| = \|\mathbf{ABx}\| \leq \|\mathbf{A}\|_F \, \|\mathbf{Bx}\| \quad . \tag{5.4}$$

Here, $\|\cdot\|_F$ denotes the Frobenius norm and $\|\cdot\|$ the L_2 vector norm; this inequality reflects the fact that the Frobenius norm is *compatible* with the L_2 vector norm. Note that using this approximation for the norm computation *bounds the output bound* of the DAUs and is *at least as tight* as the bound in Equation (5.3). As a result, without the bias term $\mathbf{b}$, the output of the corresponding DAUs can be calculated as

$$\text{eDAU}(\mathbf{x}) = \|\mathbf{Bx}\|^{-1} \, (\mathbf{Bx})^T \left(\mathbf{A}'^T\mathbf{x}\right) \tag{5.5}$$

$$\text{with} \quad \mathbf{A}' = \|\mathbf{A}\|_F^{-1}\mathbf{A} \tag{5.6}$$

We will henceforth refer to this non-linear output computation as *weight bounding* (WB). Note that under this formulation the d-dimensional weights $\mathbf{w}(\mathbf{x})$ are never explicitly calculated and the output is instead obtained as a dot product in $\mathbb{R}^r$ between the vectors $\mathbf{Bx}$ and $\mathbf{A}'^T\mathbf{x}$. Further, for convolutional DAUs (see Section 5.1.4), the matrix $\mathbf{A}'$ has to be computed only once for all positions. As we show in Section 5.3.3, this can result in significant gains in efficiency.

5.1.3 DAUs for Classification

In a classification task, one can apply k DAUs in parallel to obtain an output $\hat{\mathbf{y}}(\mathbf{x}) = [\text{DAU}_1(\mathbf{x}), ..., \text{DAU}_k(\mathbf{x})]$. Note that this is a linear transform $\hat{\mathbf{y}}(\mathbf{x}) = \mathbf{W}(\mathbf{x})\mathbf{x}$, with each row in $\mathbf{W} \in \mathbb{R}^{k \times d}$ corresponding to the weight

to achieve maximal output.

[2] Given $r=3$, the EVs maximally span a 3-dimensional subspace.

vector $\mathbf{w}_j^T$ of a specific DAU j. Consider, for example, a dataset $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i) \,|\, \mathbf{x}_i \in \mathbb{R}^d, \mathbf{y}_i \in \mathbb{R}^k\}$ of k classes with 'one-hot' encoded labels $\mathbf{y}_i$ for the inputs $\mathbf{x}_i$. To optimise the DAUs as classifiers on $\mathcal{D}$, we can apply a sigmoid non-linearity to each DAU output and optimise the loss function $\mathcal{L} = \sum_i \text{BCE}(\sigma(\hat{\mathbf{y}}_i), \mathbf{y}_i)$, where BCE denotes the binary cross-entropy and σ applies the sigmoid function to each entry in $\hat{\mathbf{y}}_i$. Note that for a given sample, BCE either maximises (DAU for correct class) or minimises (DAU for incorrect classes) the output of each DAU. Hence, this classification loss will still maximise the (signed) cosine between the weight vectors $\mathbf{w}(\mathbf{x}_i)$ and $\mathbf{x}_i$.

To illustrate this property, in Figure 5.3 (top) we show the weights $\mathbf{w}(\mathbf{x}_i)$ for several samples of the digit '3' after optimising the DAUs for classification on a noisy MNIST dataset; the first two are correctly classified, the last one is misclassified as a '5'. As can be seen, the weights align with the respective input (the weights for different samples are different). However, different parts of the input are either positively or negatively correlated with a class, which is reflected in the weights: for example, the extended stroke on top of the '3' in the misclassified sample is assigned *negative weight* and, since the background noise is *uncorrelated* with the class labels, it is not represented in the weights.

In a classification setting, the DAUs thus preferentially encode *the most frequent discriminative patterns* in the linear transform $\mathbf{ABx} + \mathbf{b}$ such that the dynamic weights $\mathbf{w}(\mathbf{x})$ align well with these patterns. Further, since the output for class j is a linear transformation of the input (**P1**), we can compute the contribution vector $\mathbf{s}_j$ containing the per-pixel contributions to this output by the element-wise product ($\odot$)

$$\mathbf{s}_j(\mathbf{x}_i) = \mathbf{w}_j(\mathbf{x}_i) \odot \mathbf{x}_i \quad , \tag{5.7}$$

see Figure 5.1 and 5.3. Such linear decompositions constitute the model-inherent 'explanations' which we evaluate in Section 5.3.

5.1.4 Convolutional Dynamic Alignment Networks

The modelling capacity of a single layer of DAUs is limited, similar to a single linear classifier. However, DAUs can be used as the basic building block for deep convolutional neural networks, which yields powerful classifiers. Importantly, in this section we show that such a Convolutional Dynamic Alignment Network (CoDA Net) inherits the properties (**P3**) of the DAUs by maintaining both the dynamic linearity (**P1**) and the alignment maximisation (**P2**). For a *convolutional* dynamic alignment layer, each convolutional filter is modelled by a DAU, similar to dynamic local filtering layers [JDBTG16]. Note that the output of such a layer is also a dynamic linear transformation of the input to that layer, since a convolution is equivalent to a linear layer with certain constraints on the weights, cf. [Sha18]. We include the implementation details in Appendix B. Finally, at the end of this section, we highlight an important difference between output maximisation and optimising for classification with the BCE loss. In this context we discuss the effect of *temperature scaling* and the loss function we optimise in our experiments.

Dynamic Linearity (P1). In order to see that the linearity is maintained, we note that the successive application of multiple layers of DAUs also results in a dynamic linear mapping. Let $\mathbf{W}_l$ denote the linear transformation matrix produced by a layer of DAUs and let $\mathbf{a}_{l-1}$ be the input vector to that layer; as mentioned before, each row in the matrix $\mathbf{W}_l$ corresponds to the weight vector of a single DAU[3]. As such, the output of this layer is given by

$$\mathbf{a}_l = \mathbf{W}_l(\mathbf{a}_{l-1})\mathbf{a}_{l-1} \quad . \tag{5.8}$$

[3]Note that this also holds for convolutional DAUs. Specifically, each row in the matrix $\mathbf{W}_l$ corresponds to a single DAU applied to exactly one spatial location in the input and the input with spatial dimensions is vectorised to yield $\mathbf{a}_{l-1}$. For details, we kindly refer the reader to [Sha18] and the implementation details in Appendix B of this work.

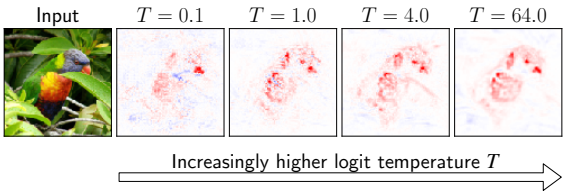


Figure 5.5: **Contribution maps for a model trained with different temperatures.** By lowering the upper bound (cf. Equation (5.3)), the alignment maximisation in the DAUs can be emphasised.

In a network of DAUs, the successive linear transformations can thus be collapsed. I.e., *for any pair of activation vectors* $\mathbf{a}_{l_1}$ and $\mathbf{a}_{l_2}$ with $l_1 < l_2$, $\mathbf{a}_{l_2}$ can be expressed as a linear transformation of $\mathbf{a}_{l_1}$:

$$\mathbf{a}_{l_2} = \mathbf{W}_{l_1 \to l_2}(\mathbf{a}_{l_1})\,\mathbf{a}_{l_1} \tag{5.9}$$

$$\text{with} \quad \mathbf{W}_{l_1 \to l_2}(\mathbf{a}_{l_1}) = \textstyle\prod_{k=l_1+1}^{l_2} \mathbf{W}_k(\mathbf{a}_{k-1}) \quad . \tag{5.10}$$

For example, the matrix $\mathbf{W}_{0 \to L}(\mathbf{a}_0 = \mathbf{x}) = \mathbf{W}(\mathbf{x})$ models the linear transformation from the input to the output space, see Figure 5.1. Since this linearity holds between any two layers, the j-th entry of any activation vector $\mathbf{a}_l$ in the network can be decomposed into input contributions via:

$$\mathbf{s}_j^l(\mathbf{x}_i) = [\mathbf{W}_{0 \to l}(\mathbf{x}_i)]_j^T \odot \mathbf{x}_i \quad , \tag{5.11}$$

with $[\mathbf{W}]_j$ the j-th row in the matrix. E.g., the contribution map for class c is given by $\mathbf{s}_c^L(\mathbf{x}_i)$ with L the model depth.

NB: In practice, we can easily extract the effective linear weighting vector $[\mathbf{W}_{0 \to l}(\mathbf{x}_i)]_j$ via a single backward pass through the model. For this to yield the weight vector $[\mathbf{W}_{0 \to l}(\mathbf{x}_i)]_j$, it is sufficient to ensure that the intermediate weight matrices $\mathbf{W}_l(\mathbf{a}_{l-1})$ are 'detached' from the computational graph such that they are treated as static (not dependent on their inputs) in the gradient computation. Specifically, to obtain $[\mathbf{W}_{0 \to l}(\mathbf{x}_i)]_j$, we calculate the gradient of the j-th unit in the l-th layer with respect to the input $\mathbf{x}_i$.

Alignment Maximisation (P2). Note that the output of a CoDA Net is bounded independent of the network parameters: since each DAU operation can—independent of its parameters—at most reproduce the norm of its input (Equation (5.3)), the linear concatenation of these operations necessarily also has an upper bound which does not depend on the parameters. Therefore, in order to achieve maximal outputs on average (e.g., the class logit over the subset of images of that class), all DAUs in the network need to produce weights $\mathbf{w}(\mathbf{a}_l)$ that align well with the class features. In other words, the weights will align with discriminative patterns in the input. For example, in Figure 5.3 (bottom), we visualise the 'global matrices' $\mathbf{W}_{0 \to L}$ and the corresponding contributions (Equation (5.11)) for a $L = 5$ layer CoDA Net. As before, the weights align with discriminative patterns in the input and do not encode the uninformative noise.

Temperature Scaling and Loss Function. So far we have assumed that minimising the BCE loss for a given sample is equivalent to applying a maximisation or minimisation loss to the individual outputs of a CoDA Net. While this is in principle correct, BCE introduces an additional, non-negligible effect: *saturation*. Specifically, it is possible for a CoDA Net to achieve a low BCE loss without the need to produce well-aligned weight vectors. As soon as the classification accuracy is high and the outputs of the networks are large, the gradient—and therefore the *alignment pressure*—will vanish. This effect can, however, easily be mitigated: as discussed in the previous paragraph, the output of a CoDA Net is upper-bounded *independent of the network parameters*, since each DAU in the network is upper-bounded. By scaling the network output with a temperature parameter T such that $\hat{\mathbf{y}}(\mathbf{x}) = T^{-1}\mathbf{W}_{0 \to L}(\mathbf{x})\,\mathbf{x}$, we can explicitly decrease this upper bound and thereby increase the *alignment pressure* in the DAUs by avoiding the early saturation due to BCE. In particular, the lower the upper bound is, the stronger the induced DAU output maximisation should be, since the network needs to accumulate more signal to obtain

large class logits (and thus a negligible gradient). This is indeed what we observe both qualitatively, cf. Figure 5.5, and quantitatively, cf. Figure 5.6 (right column). The overall loss for an input $\mathbf{x}_i$ and the target vector $\mathbf{y}_i$ is thus computed as

$$\mathcal{L}(\mathbf{x}_i, \mathbf{y}_i) = \text{BCE}(\sigma(T^{-1}\mathbf{W}_{0\to L}(\mathbf{x}_i)\,\mathbf{x}_i + \mathbf{b}_0)\,,\,\mathbf{y}_i) \quad . \tag{5.12}$$

Here, σ applies the sigmoid activation to each vector entry and $\mathbf{b}_0$ is a fixed bias term. As an alternative to the temperature scaling, the explicit representation of the network's computation as a linear mapping allows to directly regularise what properties these mappings should fulfill. For example, we show in Appendix B that by regularising the absolute values of the matrix $\mathbf{W}_{0\to L}$, we can induce sparsity in the weight alignments, leading to sharper heatmaps.

5.2 Experimental Setup

5.2.1 Datasets

We evaluate and compare the accuracies of the CoDA Nets to other work on the CIFAR10 [Kri09] and the TinyImageNet [Joh16] datasets. We use the same datasets for the quantitative evaluations of the model-inherent contribution maps. Additionally, we qualitatively show high-resolution examples from a CoDA Net trained on the first 100 classes of the ImageNet [DDS+09] dataset. Lastly, we evaluate hybrid models (see Section 5.2.4) on CIFAR10 and the full ImageNet dataset, both in terms of interpretability and classification accuracy.

5.2.2 Models

Our results (Section 5.3.1–5.3.3) are based on models of various sizes denoted by (S/L/XL)-CoDA on CIFAR10 (S), ImageNet-100 (L), and TinyImageNet (XL); these models have 7-8M[4] (S), 48M (L), and 62M (XL) parameters; see Appendix B for architecture details, an evaluation of the impact of model size on accuracy, and convergence curves. For the hybrid networks (Sections 5.2.4 and 5.3.4), we use a ResNet-56 (ResNet-50) as a base model on CIFAR10 (ImageNet) and train CoDA Nets on feature maps extracted at different depths of those models; see Appendix B for details.

5.2.3 Input Encoding

In Section 5.1.1, we discussed that the norm-weighted cosine similarity between the dynamic weights and the layer inputs is optimised and the output of a DAU is at most the norm of its input. When using pixels as the input to the CoDA Nets, this favours pixels with large RGB values, since these have a larger norm and can thus produce larger outputs in the maximisation task. We explore two approaches to mitigate this bias: in the first, we add the negative image as three additional colour channels and thus encode each pixel in the input as $[r, g, b, 1-r, 1-g, 1-b]$, with $r, g, b \in [0, 1]$.

Secondly, we show that it is also possible to train CoDA Nets on end-to-end optimised patch-embeddings and obtain similar performance in terms of interpretability and classification accuracy. Instead of computing the *per-pixel* contributions to assess the importance of spatial locations (cf. Equation (5.11)), in this setting we decompose the output with respect to the contributions from the learnt embeddings via

$$\mathbf{s}_j^L(\mathbf{x}_i) = [\mathbf{W}_{0\to L}(E(\mathbf{x}_i))]_j^T \odot E(\mathbf{x}_i) \quad , \tag{5.13}$$

[4]The SQ and L2 models have 7.8M, and the WB models 7.1M (without embedding) and 7.2M (with embedding) parameters.

with $E(\cdot)$ denoting the applied embedding function and L the number of CoDA layers in the network.

5.2.4 Interpolating between Networks

Training CoDA Nets on learnt patch-embeddings (see Section 5.2.3) naturally raises the question of how complex the embedding function should be and how large its receptive field. In particular, are *pixel-wise* importance values more useful than importance values for embeddings of patches of size 3×3? How about 7×7 or 64×64? Of course, there is no single answer to this question and the 'optimal' complexity of the embedding model depends on the dataset, the task, and, ultimately, on the preferences of the end-user of such a model: for example, if a more complex embedding allows for more performant classifiers, one might wish to trade off model interpretability against model accuracy. In order to better understand such trade-offs, we propose to 'interpolate' between a conventional CNN and the CoDA Nets and investigate how this affects both model interpretability and model performance. Specifically, starting from a pre-trained CNN, we successively replace an increasing number of the later layers of the base model by CoDA layers. As such, the model output can be decomposed into *contributions coming from spatially arranged embeddings* computed by the truncated CNN model, which can give insights into how the embeddings are used to produce the classification results.

5.2.5 Additional Details

Visualisation Details. All model-inherent contribution maps (cf. Equation (5.7)) are visualised using a blue-white-red colour map over the interval $[-v, +v]$ with blue (red) denoting negative (positive) values; the contribution per pixel is given by the sum over the contributions of the individual colour channels. In particular, for all ImageNet examples, the same v is used[5], such that all of the model-inherent explanations in Figures 5.2, 5.7, 5.8, 5.10 and 5.11 use the same scale and can thus be compared *across images*.

Shared Matrix B. In our experiments, we observed that rescaling the weight vectors of the DAUs explicitly according to Equation (5.2) resulted in long training times and high memory usage. To mitigate this, we opted to share the matrix $\mathbf{B}$ between all DAUs in a given layer when using the L2 or SQ non-linearity. This increases efficiency by having the DAUs share a common r-dimensional subspace and still fixes the maximal rank of each DAU to the chosen value of r. In contrast, networks with the WB non-linearity (see eDAUs in Equation (5.5)) are specifically designed to lower the computational costs of the DAUs and are easier to train. Therefore, for CoDA Nets built with eDAUs, we do not share the matrices $\mathbf{B}$ between the eDAUs. As the inputs are thus not restricted to a common low-dimensional subspace, we expect this to increase the modelling capacity of the CoDA Nets.

5.3 EXPERIMENTS

In Section 5.3.1 we assess the classification performance of the CoDA Nets. Further, in Section 5.3.2 we evaluate the model-inherent contribution maps derived from $\mathbf{W}_{0 \to L}$ (cf. Equation (5.13)) of a CoDA Net and compare them both *quantitatively* (cf. Figure 5.6) as well as *qualitatively* (cf. Figure 5.7) to other attribution methods. Additionally, in Section 5.3.3, we discuss the impact of the different rescaling methods (cf. Equations (5.2) and (5.5)) on model interpretability and evaluation speed. Lastly, in Section 5.3.4 we investigate the hybrid models discussed in Section 5.2.4 and analyse how the depth of the embedding

[5] v is chosen as the 99.99th percentile of the absolute values in the contribution maps of the 3 most confident predictions for each class.

CIFAR10			TinyImagenet		
Model	**#params.**	**Acc. (%)**	**Model**	**#params.**	**Acc. (%)**
SENNs [AJ18]	?	78.5	ResNet-34 [Sun16]	22M	52.0
DE-CapsNet [JH20]	11M	93.0	VGG-16 [Coa17]	138M	52.2
VGG-19 [LLLZ20]	20M	93.4	+ augmentation		56.4
ResNet-110 [HZRS16]	2M	93.6	IRRCNN [AHY+20]	15M	52.2
DenseNet [HLVDMW17]	15M	94.8	ResNet-110 [TJM20]	2M	55.6
WRN-40-4 [ZK16]	9M	95.0			
S-CoDA-SQ	8M	93.2	XL-CoDA-SQ	62M	54.4
S-CoDA-L2	8M	93.0	+ augmentation		58.4
S-eCoDA-WB	7M	94.0			
+ $E(\mathbf{x})$		94.1			

Table 5.1: **Accuracy on CIFAR10 and TinyImageNet**. Results taken from specified references. The prefix of the CoDAs indicates model size, the suffix the non-linearity used (cf. Equations (5.2) and (5.5)). $E(\mathbf{x})$ denotes that a learnt embedding was used (see Section 5.2.3). Note that the model size of the SENNs is not reported in [AJ18].

function $E(\mathbf{x})$ affects the model interpretability at different depths of the resulting hybrid architectures.

5.3.1 Model Performance

Classification Performance. In Table 5.1 we compare the performances of our CoDA Nets to several other published results. Note that the referenced numbers are meant to be used as a gauge for assessing the CoDA Net performance and do not exhaustively represent the state of the art. In particular, we would like to highlight that the CoDA Net performance matches that of VGG [SZ15] and ResNet [HZRS16] models on both datasets. Additionally, we list the reported results of the SENNs [AJ18] and the DE-CapsNet [JH20] architectures for CIFAR10. Similar to our CoDA Nets, the SENNs were designed to improve interpretability by modelling the network output as a dynamic linear transformation of the input; however, while [AJ18] obtained promising results on small datasets, they are not able to compete with conventional DNN models. On the other hand, the CoDA Nets share similarities to capsule networks, which we discuss in Appendix B; to the best of our knowledge, the DE-CapsNet achieved the state of the art in the field of capsule networks on CIFAR10 when the experiments for this chapter were evaluated. As can be seen in Table 1, the CoDA Nets match the performance of these models, whilst additionally providing detailed explanations for their decisions, as we will see in the next section. Overall, we observed that the CoDA Nets deliver competitive performances that are fairly robust to the non-linearity (see Equations (5.2) and (5.5)) and the temperature (T); for an ablation study on the latter, see Appendix B. Finally, while all models achieve good classification results, we note that the WB-based CoDA Nets perform slightly better than CoDA Nets with SQ or L2 non-linearity despite having a comparable amount of parameters. As discussed in Section 5.2.5, we attribute this to the fact that for those models we do not share the matrix $\mathbf{B}$ within layers, which increases their modelling capacity.

5.3.2 Interpretability of CoDA Nets

In the following, we evaluate the model-inherent contribution maps and compare them to other commonly used attribution methods. The evaluations are based on the XL-CoDA-SQ (T$=$6400) for TinyImageNet and the S-eCoDA-WB (T$=1e6$) for CIFAR10, see Table 5.1 for the respective accuracies. The results are very similar for all three non-linearities (cf. Section 5.3.3; more results in supplement) and we just show

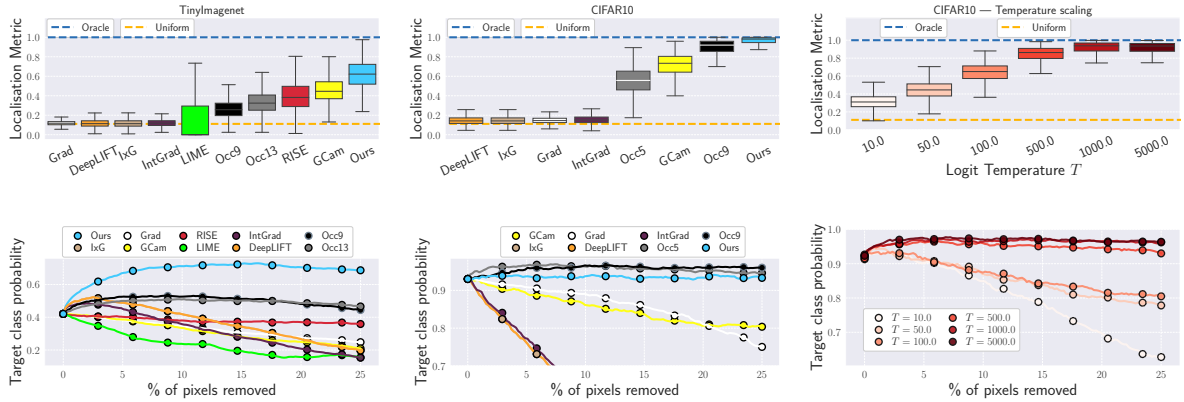


Figure 5.6: **Localisation and perturbation results.** *Top row:* Results for the localisation metric, see Equation (5.14). *Bottom row:* Pixel removal metric. In particular, we plot the mean target class probability after removing the $x\%$ of the *least important* pixels. We show the results of a CoDA-Net-SQ trained on TinyImageNet *(left column)*, as well as of a CoDA-Net-WB trained on CIFAR10 *(centre column)*. We observed the results for different rescaling methods (SQ/L2/WB) to be very similar and therefore just show one per dataset; more results can be found in Appendix B. Additionally, we show the effect of the temperature parameter on the interpretability of a CoDA Net with SQ rescaling *(right column)*: as expected, a higher temperature leads to higher interpretability (Section 5.1.4).

one per dataset as an example. Further, we evaluate the effect of training the S-CoDA-SQ architecture with different temperatures T; as discussed in Section 5.1.4, we expect the interpretability to *increase* along with T, as for larger T a stronger alignment is required in order for the models to obtain large class logits. Lastly, in Section 5.3.3 we compare how the different non-linearities (L2, SQ, WB) affect interpretability. Before turning to the results, however, in the following we first present the attribution methods used for comparison and discuss the evaluation metrics employed for quantifying their interpretability.

Attribution Methods. We compare the model-inherent contribution maps (cf. Equation (5.11)) to other common approaches for importance attribution. In particular, we evaluate against several perturbation based methods such as RISE [PDS18], LIME [RSG16], and several occlusion attributions [ZF14] (Occ-K, with K the size of the occlusion patch). Additionally, we evaluate against common gradient-based methods. These include the gradient of the class logits with respect to the input image [BSH+10] (Grad), 'Input×Gradient' (IxG, cf. [AGM+18]), GradCAM [SCD+17] (GCam), Integrated Gradients [STY17] (IntG), and DeepLIFT [SGK17]. As a baseline, we also evaluated these methods on a pre-trained ResNet-56 [HZRS16] on CIFAR10, for which we show the results in Appendix B.

Evaluation Metrics. Our quantitative evaluation of the attribution maps is based on the following two methods: we **(1)** evaluate a localisation metric by adapting the pointing game [ZBL+18] to the CIFAR10 and TinyImageNet datasets, and **(2)** analyse the model behaviour under the pixel removal strategy employed in [SF19]. For **(1)**, we evaluate the attribution methods on a grid of $n \times n$ with $n{=}3$ images sampled from the corresponding datasets; in every grid of images, each class may occur at most once. For a visualisation with $n{=}2$, see Figure 5.10. For each occurring class, we can measure how much positive importance an attribution method assigns to the respective class image. Let $\mathcal{I}_c$ be the image for class c, then the score s_c for this class is calculated as

$$s_c = \frac{1}{Z} \sum_{p_c \in \mathcal{I}_c} p_c \quad \text{with} \quad Z = \sum_k \sum_{p_c \in \mathcal{I}_k} p_c \quad , \tag{5.14}$$

with p_c the positive attribution for class c assigned to the spatial location p. This metric has the same clear oracle score $s_c{=}1$ for all attribution methods (all positive attributions located in the correct grid image)

and a clear score for completely random attributions $s_c = 1/n^2$ (the positive attributions are uniformly distributed over the different grid images). Further, note that this metric evaluates the explanations for *different classes* on the *same* image—as such, it directly measures the class-specificity of the explanation method under evaluation. Lastly, since this metric depends on the classification accuracy of the models, we sample the first 500 (CIFAR10) or 250 (TinyImageNet) images according to their class score for the ground-truth class[6]; as all attributions are evaluated for the same model on the same set of images, this does not favour any attribution method.
For (2), we show how the model's class score behaves under the removal of an increasing amount of *least important* pixels, where the importance is obtained via the respective attribution method. Since the first pixels to be removed are typically assigned negative or relatively little importance, we expect the model to initially increase its confidence (removing pixels with *negative* impact) or maintain a similar level of confidence (removing pixels with *low* impact) if the evaluated attribution method produces an accurate ranking of the pixel importance values. Conversely, if we were to remove the *most important* pixels first, we would expect the model confidence to quickly decrease. However, as noted by [SF19], removing the most important pixels first introduces artifacts in the most important regions of the image and is therefore potentially more unstable than removing the least important pixels first. Nevertheless, the model-inherent contribution maps perform well in this setting, too, as we show in Appendix B. Lastly, in Appendix B we qualitatively show that they pass the sanity check of [AGM+18].

Quantitative Results. In Figure 5.6, we compare the contribution maps of the CoDA Nets to other attribution methods under the evaluation metrics discussed above. It can be seen that the CoDA Nets (1) perform well under the localisation metric given by Equation (5.14) and outperform all other attribution methods evaluated on the same model, both for TinyImageNet (top row, left) and CIFAR10 (top row, centre); note that we excluded RISE and LIME on CIFAR10, as the default parameters do not transfer well to this low-resolution dataset. Moreover, (2) the CoDA Nets perform well in the pixel-removal setting: the *least salient* locations according to the model-inherent contributions indeed seem to be among the least relevant for the given class score on both datasets, see Figure 5.6 (bottom row, left and centre); note that the Occ-K explanations directly estimate the impact of occluding pixels and are thus expected to perform well under this metric. Further, in Figure 5.6 (right column), we show the effect of temperature scaling on the interpretability of CoDA-SQ Nets trained on CIFAR10. The results indicate that the alignment maximisation is indeed crucial for interpretability and constitutes an important difference of the CoDA Nets to other dynamic linear networks such as piece-wise linear networks (for the interpretability results of piece-wise linear models, see Appendix B). In particular, by structurally requiring a strong alignment for confident classifications, the interpretability of the CoDA Nets forms part of the optimisation objective. Increasing the temperature increases the alignment and thus the interpretability of the CoDA Nets. While we observe a downward trend in classification accuracy when increasing T, the best model at $T = 10$ only slightly outperforms a model with $T = 1000$ (93.6% vs. 93.2%); for more details, see Appendix B.

In summary, the results show that by combining dynamic linearity with a structural bias towards an alignment with discriminative patterns, we obtain models which inherently provide an interpretable linear decomposition of their predictions. Further, given that we better understand the relationship between the intermediate computations and the optimisation of the final output in the CoDA Nets, we can emphasise model interpretability in a principled way by increasing the 'alignment pressure' via *temperature scaling*.

Qualitative Results. In Figures 5.7 and 5.8, we visualise spatial contribution maps of an L-CoDA-SQ model (trained on ImageNet-100); specifically, in Figure 5.7 we show explanations for the most confident predictions for 18 different classes and additionally show explanations for multiple instances of some of

[6] We can only expect an attribution to specifically highlight a class image if this image is correctly classified on its own. If all grid images have similarly low attributions, the localisation score will be random.

Figure 5.7: **Model-inherent contribution maps for the most confident predictions for 18 different classes**, sorted by confidence (high to low). We show positive (negative) contributions (Equation (5.11)) per pixel for the ground truth class logit in red (blue). Note that all contribution maps use the same scale (see Section 5.2.5) and differences in opacity thus directly reflect *absolute* feature importance.

the classes in Figure 5.8 to highlight the explanation consistency. Note that these contribution maps are linear decompositions of the output and the sum over these maps yields the respective class logit. Further, in Figure 5.10, we present qualitative examples of the localisation metric (cf. Figure 5.6), which show that the CoDA Net explanations are highly class-specific. In Figure 5.11, we additionally present a visual comparison to the best-performing post-hoc attribution methods; note that RISE cannot be displayed well under the same colour coding and we thus use its default visualisation. We observe that the different methods are not inconsistent with each other and roughly highlight similar regions. However, the inherent contribution maps are of much higher detail and compared to the perturbation-based methods do not require multiple model evaluations. Much more importantly, however, all the other methods are attempts at approximating the model behaviour *post-hoc*, while the CoDA Net contribution maps in Figure 5.7 are derived from the model-inherent linear mapping that is used to compute the model output. Finally, note that the alignment maximisation is crucial to achieve these highly detailed explanations and dynamic linearity on its own is not sufficient to obtain interpretable models. In particular, in Figure 5.2 we compare the model-inherent linear contribution maps of CoDA Nets to the model-inherent linear contribution maps of various piece-wise linear models[7]. In contrast to the CoDA Net contribution maps, those of the piece-wise linear models are very noisy and thus difficult to interpret.

5.3.3 Interpretability and Efficiency of L2, SQ, and WB

In the following, we discuss the effect of the normalisation (L2, SQ, WB) on interpretability and efficiency.

Model Interpretability. In Figure 5.9, we show the results of the interpretability metrics for models with different rescaling functions (L2/SQ/WB, see Equations (5.2) and (5.5)) as well as for a model trained with a learnt patch-embedding $E(\mathbf{x})$. As an embedding function, we simply apply a 3x3 convolution with 32 filters, followed by a batch normalisation layer [IS15]. For comparison to post-hoc methods evaluated on a CoDA Net, we kindly refer the reader to the centre column of Figure 5.6. As can be seen, it is possible to obtain highly interpretable models under all four settings: **(1)** the linear contributions allow to localise the class-images well (localisation metric, left) and **(2)** the models are insensitive to input

[7] Note that piece-wise linear models, as the name suggests, effectively compute a linear transformation for any given input $\mathbf{x}_i$ and can thus be 'explained' in the same way as the CoDA Nets, Equation (5.11).

Figure 5.8: **The explanations for the 4 most confident predictions of 8 classes** consistently highlight similar features across images, despite changes in viewpoint; since all explanations use the same scale (see Section 5.2.5), the opacity allows to assess feature importance across images. These features can be highly localised (eyes of the owl) or distributed (feathers of the ostriches). Additional examples can be found in Appendix B.

features[8] that are not contributing to the output as per the linear transformation matrix $\mathbf{W}_{0\to L}$. Note that for the model with a learnt patch-embedding, denoted by E–WB, we show two results for the perturbation metric. First, we 'zero out' the *embeddings* at each location ordered by their assigned importance (blue crosses). As the embeddings are the input features to the CoDA Net, the model confidence shows the expected behaviour of being insensitive to unimportant inputs. In contrast, the assigned importance values do not translate to the centre pixels of the embeddings: when zeroing out the *center pixels* according to the contributions of the patch-embeddings, the model confidence drops more quickly (see red crosses). This distinction is important to keep in mind when evaluating CoDA Nets on input embeddings, since it is easy to wrongly interpret such contribution maps. If the input to the CoDA Net is an embedding of an image patch, it depends on the embedding function how the contributions are to be distributed to the image pixels. Lastly, note that different from the centre column in Figure 5.6, the metrics are evaluated for *four different models* and are thus not comparisons between different explanation methods, but rather between different models under the same explanation. As such, the differences in the localisation metric do not necessarily show that the linear decompositions are generally better suited to explain WB-based models as compared to SQ- or L2-based models; they might instead reflect the fact that the models learnt more robust and class-specific representations, which yield both better results in the localisation task as well as higher classification accuracy.

Model Efficiency. While all three non-linearities can yield interpretable CoDA Nets, the computational

[8]For the SQ, L2, and WB model the features are pixels under the static encoding function described in 5.2.3. For the E-WB model, the input features to the CoDA Net are *learnt* patch embeddings.

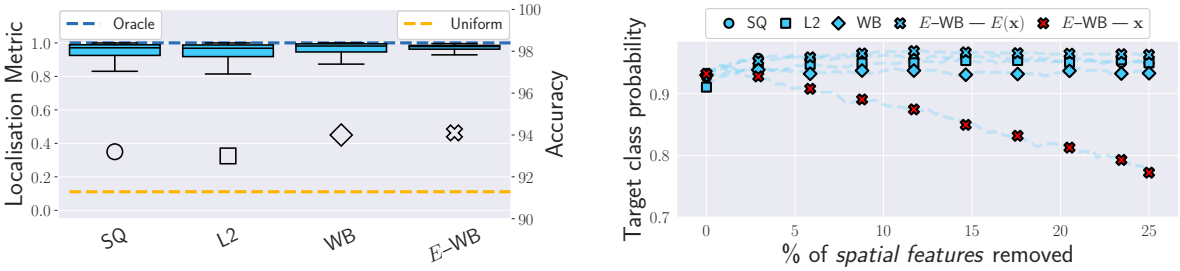


Figure 5.9: **Localisation** (*l*eft) **and perturbation** (*r*ight) **metric results** on CIFAR10 for CoDA Networks with different non-linearities (cf. Equations (5.2) and (5.5)) as well as trained with a learnt patch-embedding $E(\mathbf{x})$ (denoted E–WB). For the model with embedding $E(\mathbf{x})$, we evaluate the pixel perturbation metric (bottom) directly on the pixels (red crosses) as well as on the learnt patch-embeddings (blue crosses). We further added the models' accuracies (see Table 5.1) in the plot to the left for comparison (circle/squares/cross).

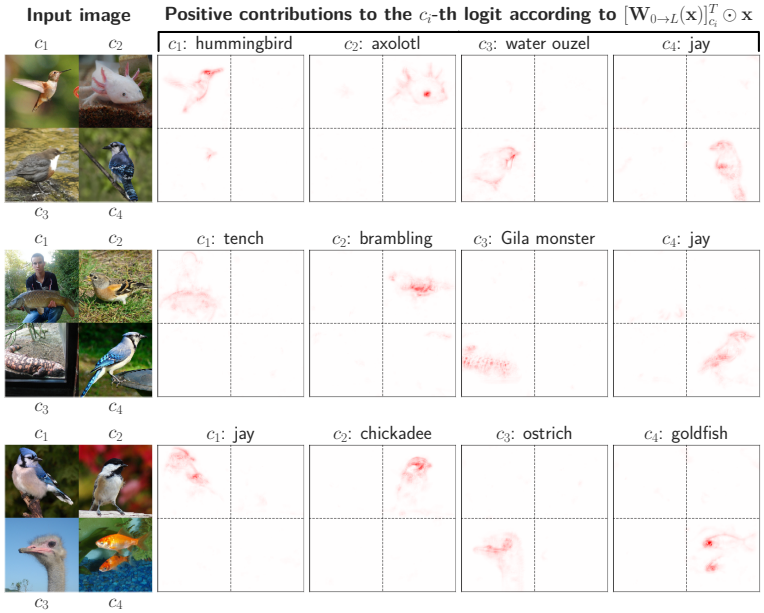


Figure 5.10: **Localisation metric.** *Col. 1:* The network is evaluated on a synthetic image consisting of n images of different classes c_i. *Cols. 2-5:* Model-inherent explanations (Equation (5.11)) for the respective model outputs (class logits c_i) as given by the CoDA Net. By evaluating on images with multiple classes, the localisation metric measures the class-specificity of the explanations.

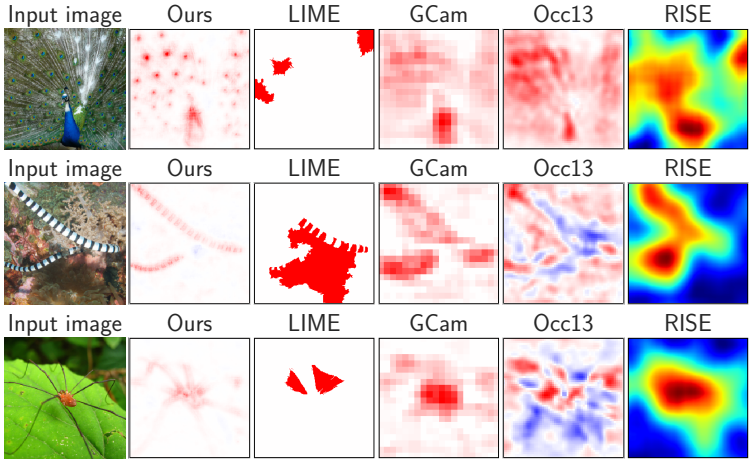


Figure 5.11: **Comparison to best post-hoc methods.** While the regions of importance roughly coincide, the inherent contribution maps of the CoDA Nets offer the most detail. To improve the RISE visualisation, we chose its default colormap [PDS18]; the most (least) important values are still shown in red (blue).

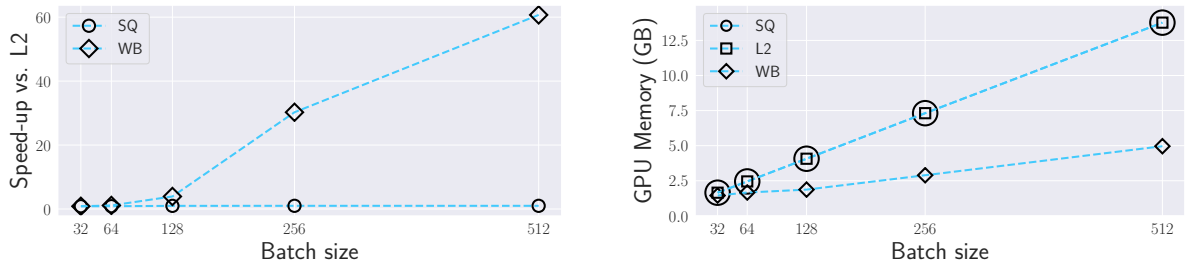


Figure 5.12: **Forward pass speed-up** of the models with WB and SQ rescaling compared to L2 (*left*) as well as the GPU memory consumption (*right*) of the respective models in Table 5.1 for different batch sizes for both measures. While the models perform similarly for small batch sizes, the WB-based model scales better to large inputs.

cost of the different approaches for bounding the DAU outputs differs. For example, by avoiding the explicit calculation of the d-dimensional weight vector in Equation (5.5), the eDAUs are able to save both memory as well as floating point operations—the computed vectors are of size $r \ll d$ and the dot-product in the low-dimensional space requires $O(r)$ operations instead of $O(d)$. Being the fundamental building block of the CoDA Nets, such gains in efficiency can have considerable impact, since the corresponding computations are performed in every layer for every unit and at each spatial position of the input to the respective layer. Accordingly, in practice we observed that the weight bounding approach in the eDAUs (Equation (5.5)) can yield significant speed-ups and memory savings, especially for high-dimensional inputs. For example, in Figure 5.12 we plot the memory consumption and forward-pass speeds for the three different models without learnt embedding function (see Table 5.1) for varying batch sizes on the CIFAR10 dataset: while SQ and L2 perform similarly, the WB-based model scales better to larger inputs.

Additionally, we measured memory consumption and training time for two models with the same architecture on ImageNet (L-CoDA, see beginning of Section 5.3) for the SQ and the WB rescaling methods. For this, we updated the models ≈ 8000 times with a batch size of 16 and recorded the overall time as well as the GPU memory consumption. In these experiments, the WB-based model required more than $3\times$ less memory (9.7GB vs. 30.0GB) and completed the updates more than $1.5\times$ faster (8.7 minutes vs. 14.1 minutes). All experiments regarding evaluation speed were performed on an nvidia Quadro RTX 8000 GPU with 48GB of memory.

5.3.4 Hybrid CoDA Networks

In this section, we assess the interpretability of hybrid CoDA Nets, which combine conventional CNN layers and CoDA layers in one network model. For our experiments, we use varying numbers of pre-trained CNN layers as feature extractors on top of which a CoDA Net is trained as a classifier. Such a hybrid structure can prove useful in cases where CoDA Nets do not (yet) yield the same accuracy as conventional architectures; for details on the network architectures we kindly refer the reader to Appendix B.

In particular, we use the first K layers of a pre-trained ResNet model [HZRS16] as feature extractors. Since ResNets are piece-wise linear models, the hybrids are still *dynamic linear* and we can assign importance values to input features according to their effective linear contribution; importantly, the input features can be extracted *at any depth* of the network as the output of a CoDA layer or a ResNet block, or as the actual input pixels. To assess whether such hybrids are more interpretable than the base model, we compute spatial contribution maps[9] with respect to different activation maps within the network and

[9]The contribution maps according to the dynamic linear mapping can be obtained via 'Input×Gradient', where for the

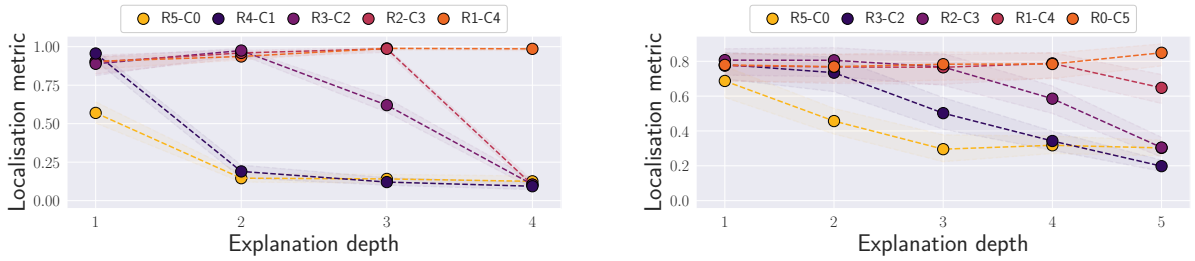


Figure 5.13: **Localisation metric (mean and standard deviation) for different 'explanation depths'** evaluated on four hybrid models trained on CIFAR10 (*left*) and ImageNet (*right*). Additionally, we show the localisation results of the pretrained ResNets (denoted by R5-C0) that were used as base models. For each evaluation, we extract the effectively applied linear transformations up to a certain depth and compute the corresponding linear contributions to the output logits coming from individual positions in the activation maps. We then use the resulting maps as an explanation of the output logit and assess how well these maps allow for localising the corresponding class images in the localisation task. As can be seen, the more layers of the original ResNet architectures are replaced by CoDA layers, the larger is the 'interpretable depth'.

evaluate them under the localisation metric (see Section 5.3.2, 'Evaluation metrics').

CIFAR10. For the following experiments, we use a pre-trained ResNet-56 obtained from [Ide18]. This model consists of a convolutional layer + batch normalisation [IS15] (C+B), followed by three times nine *residual blocks* (RBs) as well as a fully connected and a pooling (FC+P) layer; for more details we kindly refer the reader to the original work [HZRS16] and the implementation [Ide18] on which we base these experiments. We can summarise this model by [C+B, 9RB, 9RB, 9RB, FC+P]; we will further denote individual *segments* $\mathcal{S}_i$ of the model by their index in this summary counting from the back, e.g., $\mathcal{S}_5 =$ [C+B] and $\mathcal{S}_1 =$ [FC+P]. In order to evaluate the interpretability of this model at different depths t, we split it at different points into two virtual parts: an embedding function $E_t(\mathbf{x})$ and a classification head (CH_t). For a given split, we then regard the output of $E_t(\mathbf{x})$ as the input to the classification head and linearly decompose the latter according to the respective linear transformation performed by the model; e.g., with an *explanation depth* of 2 we refer to the split in which $\mathcal{S}_2$ is the first element in CH_2 and we evaluate linear contribution maps obtained for the classification head $\text{CH}_2 =$ [9RB, FC+P] on the preprocessed input $E_2(\mathbf{x}) =$ [C+B, 9RB, 9RB]$(\mathbf{x})$. By performing this evaluation for various splits, we can assess the 'interpretable depth' of a model. In particular, we evaluate how well the contribution maps at different depths allow for localising the correct class images in the localisation task.

In order to investigate the effect of CoDA layers on the interpretable depth, we train and evaluate four different hybrid models. For this, we replace an increasing number of segments $\mathcal{S}_i$ by CoDA layers, starting from $\mathcal{S}_1$; in Figure 5.13 (left), we denote the base model by R5-C0 (5 ResNet segments, 0 CoDA segments) and the hybrids according to the number of replaced segments, e.g., for R3-C2 we replaced the last two segments by CoDA layers[10]. *For each of these models*, we can decompose the model outputs in terms of contributions from spatial positions for the embedding functions E_t defined by different splits t. Note that the base model (ResNet-56) is piece-wise linear and we can thus still compute linear contributions at any depth of this hybrid network. In Figure 5.13 (left) we show the results of the localisation metric for all networks at various depths; the classification accuracies can be found in Table 5.2. As can be seen, the linear contributions are good explanations of the class logits as long as

gradient calculation we treat the dynamic matrices in the CoDA layers as fixed.

[10] In detail, each segment of 9RB is replaced by a set of 3 CoDA layers. The final network segment [FC+P] is replaced by a single CoDA layer followed by a global pooling operation.

CIFAR10	R5-C0	R4-C1	R3-C2	R2-C3	R1-C4
	93.4%	93.6%	93.4%	93.6%	93.8%
ImageNet	R5-C0	R3-C2	R2-C3	R1-C4	R0-C5
	76.1%	74.7%	73.3%	71.7%	71.4%

Table 5.2: **Classification accuracies for hybrid networks.** RX-CY denotes how many segments were replaced by CoDA layers; on ImageNet, maximally up to five ResNet blocks from the end of the network were replaced and all networks thus still rely on a ResNet-based stem. On CIFAR10 the accuracy can be maintained whilst improving interpretability (see Figure 5.13). On ImageNet, on the other hand, we observe a trade-off in accuracy when increasing the 'interpretable depth' of the models.

the classification head entirely consists of CoDA layers and drops as soon as we include a segment with ResNet blocks in the classification head CH. Again, this highlights that *dynamic linearity* alone is not enough to obtain useful linear decompositions of the model outputs, but that the alignment property is crucial for the interpretability of the CoDA layers.

ImageNet. In the following, we show that the gains from interpolating between networks also extend to a more complex dataset. Similar to the interpolation experiments on CIFAR10 above, the results are based on an interpolation between a pretrained ResNet model (ResNet-50) and a CoDA-based classification head. However, given the high-dimensional representations produced by the later ResNet layers (up to 2048 channels), the parameters of the classification head increase drastically if the high dimensionality is maintained throughout the CoDA layers. Therefore, in the ImageNet experiments, we first compute a low-dimensional projection $\tilde{\mathbf{x}}=\mathbf{P}\mathbf{x}$ of the inputs to the convolutional kernels to which we apply the eDAUs (see Equation (5.5)); similarly to the dynamic weights of DAUs with L2 normalisation, we normalise the rows of the matrix $\mathbf{P}$ to unit norm to maintain a parameter-independent bound of the network. For the interpolation experiments, we successively replace the last 5 residual blocks of the ResNet-50 base model (the 'segments' here correspond to FC+P or individual residual blocks) by a single CoDA layer each and assess the interpretability via the localisation metric as well as model accuracy, see Figure 5.13 (right) and Table 5.2 respectively. Similar to the CIFAR10 experiments, we observe an increase in 'interpretable depth' (Figure 5.13, right). However, while on CIFAR10 the accuracy of the base model could be maintained, on ImageNet we observe a trade-off in accuracy. While better results can certainly be achieved by further optimising the network architectures or fine-tuning the learnt embeddings, our results show that it is possible to increase the interpretability of performant classification models by using a classification head comprised of CoDA layers.

5.4 CONCLUSION

We presented a new family of neural networks, the CoDA Nets, and showed that they are performant classifiers with a high degree of interpretability. For this, we first introduced the Dynamic Alignment Units (DAUs), which model their output as a dynamic linear transformation of their input and have a structural bias towards alignment maximisation. This bias is induced by ensuring that a DAU can only produce large outputs if its weights align with the input, since the dynamically applied weights are explicitly normalised. To lower the computational costs of the DAUs, we further introduce the eDAUs, for which we normalise the weights by an upper bound of their norms which is cheaper to compute. Using the DAUs to model filters in a convolutional network, we obtain the Convolutional Dynamic Alignment Networks (CoDA Nets). The successive linear mappings by means of the DAUs within the network make it possible to linearly decompose the output into contributions from individual input dimensions—in

contrast to piece-wise linear networks, which are also dynamic linear, the alignment property of the DAUs ensures that the linear decomposition aligns with discriminant patterns in the input. In order to assess the quality of these contribution maps, see Equation (5.11), we compare against other attribution methods. We find that the CoDA Net contribution maps consistently perform well under commonly used quantitative metrics and are robust to the applied normalisation scheme. Beyond their *interpretability*, the CoDA Nets constitute performant classifiers: their accuracy on CIFAR10 and the TinyImageNet dataset are on par to the commonly employed VGG and ResNet models. Lastly, we showed that CoDA layers can be combined with conventional DNNs, which yields hybrid models with an increased 'interpretable depth' compared to the base model. As such, these hybrid models take advantage of the high modelling capacity and efficiency of modern DNNs whilst allowing for a user-defined 'minimal interpretability'. While this approach enabled us to scale the CoDA Nets to the full ImageNet dataset and improve the interpretable depth of the original models, this came at a cost in classification accuracy. In the next section, we present an alternative approach for designing interpretable DNNs that achieve competitive performance, even without the need of combining them with conventional DNNs in a hybrid manner: the B-cos Networks.

6 B-cos Alignment is All We Need for Interpretability

Contents

Similar to Chapter 5, this chapter is concerned with designing Deep Neural Networks (DNNs) that inherently provide explanations that constitute a faithful model summary *and* have a clear interpretation for humans. For this, we take advantage of the same ingredients that we successfully applied in the CoDA Networks in the previous chapter: dynamic linearity and alignment pressure.

Additionally, in this chapter we place a particular focus on scalability and extend the benefits observed for the CoDA Networks to state-of-the-art DNN architectures. Crucially, we show that the resulting models not only provide high-quality explanations for their decisions, see Figure 6.1, but even maintain most of their classification performance on the challenging ImageNet [DDS+09] classification dataset.

To achieve this, we manipulate one of the most fundamental operations in modern DNNs and design the proposed **B-cos transformation** as a drop-in replacement for linear transformations. As a result, the B-cos transformation can easily be integrated into a wide range of existing DNN architectures. Importantly, the B-cos transformation is a dynamic linear operation and inherently biased towards weight-input alignment. Similar to the CoDA Networks, the resulting models are thus also dynamic linear and can be summarised faithfully by a single linear transformation for any given input. Moreover, as the model weights align with task-relevant patterns in the input during optimisation, these summarising linear transformations become easily interpretable for humans: they are a direct reflection of the weights the model has learnt during training and specifically reflect those weights that best align with a given input.

In summary, in this chapter we make the following contributions:

(1) We introduce the B-cos transformation to improve the interpretability of DNNs. By promoting weight-input alignment, these transformations are explicitly designed to yield explanations that highlight task-relevant input patterns.

(2) Specifically, the B-cos transformation is designed such that any sequence of B-cos transformations can be faithfully summarised by a single linear transformation. We show that this allows to explain not only the models' outputs, but also representations in intermediate network layers.

Figure 6.1: **Explanation examples of a B-cos DenseNet-121.** *Top:* Input images $\mathbf{x}_i$ to the model. *Bottom:* B-cos explanation for class c (c: image label). Specifically, we visualise the c-th row of $\mathbf{W}_{1\to L}(\mathbf{x}_i)$ as applied by the model, see Equation (6.14); no masking of the original image is used. For the last 2 images, we also show the explanation for the 2nd most likely class. For details on visualising $\mathbf{W}_{1\to L}(\mathbf{x}_i)$, see Section 6.2.

(3) We demonstrate that a plain B-cos convolutional neural network without any additional non-linearities, normalisation layers or regularisation schemes achieves competitive performance on CIFAR10 [Kri09], demonstrating the modelling capacity of DNNs that are solely based on the B-cos transformation. In this context, we show that the parameter B gives fine-grained control over the increase in weight alignment and thus the interpretability of the B-cos Networks.

(4) We analyse how to integrate normalisation layers into B-cos models to take advantage of the optimisation benefits of those layers without sacrificing interpretability.

(5) We analyse how to combine the proposed B-cos framework with attention-based models (e.g. Vision Transformers (ViTs) [DBK+21]), which highlights the generality of our approach and shows that it extends beyond convolutional networks.

(6) Finally, we show in a wide range of experiments that B-cos DNNs achieve similar accuracy as their conventional counterparts. Specifically, we evaluate the B-cos versions of various commonly used DNNs such as VGGs [SZ15], ResNets [HZRS16], DenseNets [HLVDMW17], ResNeXt [XGD+17] and ConvNeXt models [LMW+22], as well as ViTs [DBK+21]. Our strongest models achieve >80% top-1 accuracy on ImageNet, all while providing highly detailed explanations for their decisions. Specifically, the B-cos explanations outperform other explanation methods across all tested architectures, both under quantitative metrics as well as under qualitative inspection.

This chapter is based on [BFS22b, BSFS24] and the corresponding code is publicly available at: github.com/B-cos/B-cos-v2.

6.1 B-cos Neural Networks

In this section, we introduce the B-cos transformation as a replacement for the linear units in DNNs, which are (almost) "at the heart of every deep network" [RZL18], and discuss how this can increase the interpretability of DNNs.

For this, we first introduce the B-cos transformation as a variation of the linear transformation (6.1.1) and highlight its most important properties. Then, we show how to construct B-cos Networks (6.1.2) and how to faithfully summarise the network computations to obtain explanations for their outputs (6.1.2.1). Subsequently, we discuss how the B-cos transformation—combined with the binary cross entropy (BCE) loss—affects the parameter optima of the models (6.1.2.2). Specifically, by inducing alignment pressure, the B-cos transformation aligns the model weights with task-relevant patterns in the input. Finally, in Section 6.1.3 we integrate the B-cos transformation into conventional DNNs by using it as a drop-in replacement for the ubiquitously used linear units and discuss how to combine B-cos layers with normalisation and attention layers without sacrificing the interpretability of B-cos Networks.

6.1.1 The B-cos Transformation

Typically, the individual 'neurons' in a DNN compute the dot product between their weights $\mathbf{w}$ and an input $\mathbf{x}$:

$$f(\mathbf{x};\mathbf{w}) = \mathbf{w}^T\mathbf{x} = \|\mathbf{w}\|\,\|\mathbf{x}\|\,c(\mathbf{x},\mathbf{w})\,, \tag{6.1}$$

$$\text{with}\quad c(\mathbf{x},\mathbf{w}) = \cos\left(\angle(\mathbf{x},\mathbf{w})\right)\,. \tag{6.2}$$

Here, $\angle(\mathbf{x},\mathbf{w})$ returns the angle between the vectors $\mathbf{x}$ and $\mathbf{w}$. As we seek to improve the interpretability of DNNs by promoting weight-input alignment during optimisation, we propose to explicitly emphasise the alignment term in the linear transformation, i.e. the impact of the cosine function. This yields the **B-cos transformation**:

$$\text{B-cos}(\mathbf{x};\mathbf{w}) = \underbrace{\|\widehat{\mathbf{w}}\|}_{=1}\,\|\mathbf{x}\|\,|c(\mathbf{x},\widehat{\mathbf{w}})|^{\text{B}} \times \operatorname{sgn}\left(c(\mathbf{x},\widehat{\mathbf{w}})\right)\,. \tag{6.3}$$

In Equation (6.3), the hat-operator scales $\widehat{\mathbf{w}}$ to unit norm, B is a scalar, and sgn the sign function. The B-cos transformation thus rescales the output of a linear transformation and equals:

$$\text{B-cos}(\mathbf{x};\mathbf{w}) = \widehat{\mathbf{w}}^T\mathbf{x} \times |\cos(\mathbf{x},\widehat{\mathbf{w}})|^{\text{B-1}}\,; \tag{6.4}$$

for a derivation of the equivalence, see Section C.4.2 in the appendix.

Note that this only introduces *minor changes* (highlighted in blue) with respect to Equation (6.1); e.g., for $\text{B}=1$, Equation (6.4) is equivalent to a linear transformation with $\widehat{\mathbf{w}}$. However, albeit small, these changes are important for three reasons.

First, similar to the DAUs in the preceding chapter, the output of the B-cos transformation is bounded:

$$\|\widehat{\mathbf{w}}\| = 1 \;\Rightarrow\; \text{B-cos}(\mathbf{x};\mathbf{w}) \leq \|\mathbf{x}\|\,. \tag{6.5}$$

Importantly, equality in Equation (6.5) implies that $\mathbf{x}$ and $\mathbf{w}$ are collinear, i.e., *aligned* (cf. Equation (6.3)).

Secondly, an increased exponent B suppresses the outputs for weight-input pairs with low cosine similarity,

$$\text{B} \gg 1 \wedge |c(\mathbf{x},\widehat{\mathbf{w}})| < 1 \;\Rightarrow\; \text{B-cos}(\mathbf{x};\mathbf{w}) \ll \|\mathbf{x}\|\,, \tag{6.6}$$

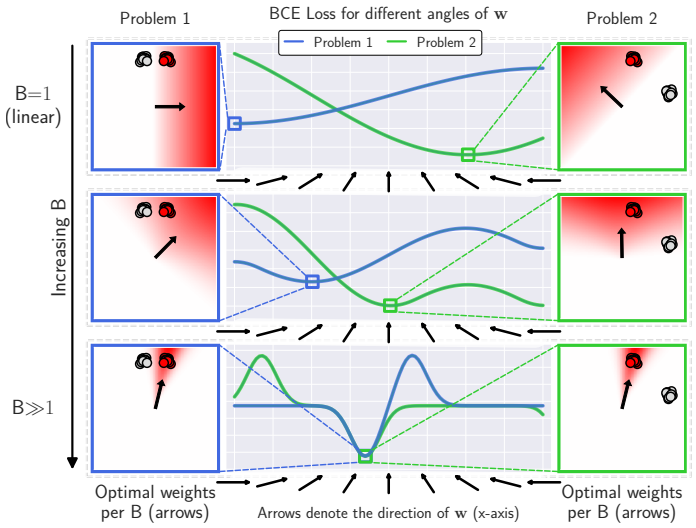


Figure 6.2: **B-cos classification example.** *Col. 2:* BCE loss for different angles of $\mathbf{w}$ for B-cos classifiers (Equation (6.3)) with different B (rows) for two classification problems. *Cols. 1+3:* Visualisation of the classification problems and the corresponding optimal weights (arrows) per B. For $\mathrm{B}=1$ (first row) the weights $\mathbf{w}$ represent the decision boundary of a linear classifier. Although the red cluster is the same in both cases, the optimal weight vectors differ significantly (compare within row). In contrast, for higher B the weights converge to the same optimum in both tasks (see last row). The opacity of the red shading shows the strength of the positive activation of the B-cos transformation for a sample at a given position.

and the respective B-cos unit only yields outputs close to its maximum (i.e. $\|\mathbf{x}\|$) for a small range of angular deviations from $\mathbf{x}$. Together, these two properties can significantly change the optimal parameters. To illustrate this, we show in Figure 6.2 how increasing B affects a simple linear classification problem. In particular, note how increasing B narrows the 'response window' (red area) of the B-cos transformation and suppresses the influence of the distractors (grey circles). In contrast to a linear classifier, which has a highly task-dependent optimum (cf. first row in Figure 6.2), the B-cos transformation thus classifies based on *cosine similarity* between weights and inputs. As such, the optimal weights constitute 'angular prototypes' and lend themselves well for explaining the model prediction: a sample is confidently classified as the red class if it 'looks like' the weight vector.

Lastly, the B-cos transformation maintains an important property of the linear transformation: specifically, sequences of B-cos transformations are faithfully summarised by a single linear transformation (Equation (6.14)). As a result, the explanations for individual neurons in a DNN can easily be combined to yield explanations for the full model, as we discuss in Section 6.1.2.1. Importantly, these explanations will align with discriminative patterns when optimising a B-cos Network for classification (Section 6.1.2.2), making them easily interpretable for humans.

6.1.2 Simple (convolutional) B-cos Networks

In this section, we discuss how to construct simple (convolutional) DNNs based on the B-cos transformation. Then, we show how to summarise the network outputs by a single linear transformation and, finally, why this transformation aligns with discriminative patterns in classification tasks.

B-cos Networks. The B-cos transformation is designed to replace the linear transformation and can be used in exactly the same way. E.g., consider a *conventional* fully connected multi-layer neural network $\mathbf{f}(\mathbf{x};\theta)$ of L layers, defined as

$$\mathbf{f}(\mathbf{x};\theta) = \mathbf{l}_L \circ \mathbf{l}_{L-1} \circ \ldots \circ \mathbf{l}_2 \circ \mathbf{l}_1(\mathbf{x}) \;, \tag{6.7}$$

with $\mathbf{l}_j$ denoting layer j with parameters $\mathbf{w}_j^k$ for neuron k in layer j, and θ the collection of all model parameters. In such a model, each layer $\mathbf{l}_j$ typically computes

$$\mathbf{l}_j(\mathbf{a}_j;\mathbf{W}_j) = \phi\left(\mathbf{W}_j\,\mathbf{a}_j\right) \;, \tag{6.8}$$

with $\mathbf{a}_j$ the input to layer j, ϕ a non-linear activation function (e.g., ReLU), and the row k of $\mathbf{W}_j$ given by the weight vector $\mathbf{w}_j^k$ of the k-th neuron in that layer. Note that the non-linear activation function ϕ is *required* to be able to model non-linear relationships with multiple layers.
A corresponding B-cos Network $\mathbf{f}^*$ with layers $\mathbf{l}_j^*$ can be formulated in exactly the same way as

$$\mathbf{f}^*(\mathbf{x};\theta) = \mathbf{l}_L^* \circ \mathbf{l}_{L-1}^* \circ ... \circ \mathbf{l}_2^* \circ \mathbf{l}_1^*(\mathbf{x}) \,, \tag{6.9}$$

with the only difference being that every dot product (here between rows of $\mathbf{W}_j$ and inputs $\mathbf{a}_j$) is replaced by the B-cos transformation in Equation (6.4). In matrix form, this equates to

$$\mathbf{l}_j^*(\mathbf{a}_j;\mathbf{W}_j) = \left(\widehat{\mathbf{W}}_j\,\mathbf{a}_j\right) \times |c(\mathbf{a}_j;\widehat{\mathbf{W}}_j)|^{\mathrm{B}-1} \,. \tag{6.10}$$

Here, the power, absolute value, and $\times$ operators are applied element-wise, $c(\mathbf{a}_j;\widehat{\mathbf{W}}_j)$ computes the cosine similarity between input $\mathbf{a}_j$ and the rows of $\widehat{\mathbf{W}}_j$, and the hat operator scales the rows of $\widehat{\mathbf{W}}_j$ to unit norm. Finally, as for $\mathrm{B}>1$ the transformation $\mathbf{l}_j^*$ is already *non-linear*, a non-linear ϕ is not required for modelling non-linear relationships.

The above discussion readily generalises to convolutional neural networks (CNNs): in CNNs, we replace the linear transformations computed by the convolutional kernels by B-cos, see Algorithm C.1 in Appendix C. Although we assumed a plain multi-layer network without add-ons (e.g. skip connections), we show in Section 6.3 that the benefits of B-cos also transfer to more advanced architectures (Section 6.1.3).

6.1.2.1 *Computing Explanations for B-cos Networks*

As can be seen by rewriting Equation (6.10), a B-cos layer effectively computes an input-dependent linear transformation:

$$\mathbf{l}_j^*(\mathbf{a}_j;\mathbf{W}_j) \;=\; \widetilde{\mathbf{W}}_j(\mathbf{a}_j)\,\mathbf{a}_j \,, \tag{6.11}$$

$$\text{with} \quad \widetilde{\mathbf{W}}_j(\mathbf{a}_j) \;=\; |c(\mathbf{a}_j;\widehat{\mathbf{W}}_j)|^{\mathrm{B}-1} \odot \widehat{\mathbf{W}}_j \,. \tag{6.12}$$

Here, $\odot$ scales the rows of the matrix to its right by the scalar entries of the vector to its left. Hence, the output of a B-cos Network, see Equation (6.9), is effectively calculated as

$$\mathbf{f}^*(\mathbf{x};\theta) = \widetilde{\mathbf{W}}_L(\mathbf{a}_L)\widetilde{\mathbf{W}}_{L-1}(\mathbf{a}_{L-1})...\widetilde{\mathbf{W}}_1(\mathbf{a}_1{=}\mathbf{x})\mathbf{x} \,. \tag{6.13}$$

As multiple linear transformations in sequence can be collapsed to a single one, $\mathbf{f}^*(\mathbf{x};\theta)$ can be written as

$$\mathbf{f}^*(\mathbf{x};\theta) = \mathbf{W}_{1\to L}(\mathbf{x})\,\mathbf{x} \,, \tag{6.14}$$

$$\text{with} \quad \mathbf{W}_{1\to L}\,(\mathbf{x}) = \textstyle\prod_{j=1}^{L} \widetilde{\mathbf{W}}_j\,(\mathbf{a}_j) \;. \tag{6.15}$$

Thus, $\mathbf{W}_{1\to L}(\mathbf{x})$ faithfully summarises the network computations (Equation (6.9)) by a single linear transformation (Equation (6.14)).

To explain an activation (e.g., the class logit), we can now either directly visualise the corresponding row in $\mathbf{W}_{1\to L}$, see Figures 6.1 and 6.10, or the *contributions* according to $\mathbf{W}_{1\to L}$ coming from individual input dimensions. We use the resulting spatial contributions maps to quantitatively evaluate the explanations. In detail, the input contributions $\mathbf{s}_j^l(\mathbf{x})$ to neuron j in layer l for an input $\mathbf{x}$ are given by the

$$\textbf{contribution map} \qquad \mathbf{s}_j^l(\mathbf{x}) = [\mathbf{W}_{1\to l}(\mathbf{x})]_j^T \odot \mathbf{x} \,, \tag{6.16}$$

with $[\mathbf{W}_{1\to l}]_j$ denoting the jth row in matrix $\mathbf{W}_{1\to l}$; as such, the contribution from a single pixel location (x, y) is given by $\sum_c [\mathbf{s}_j^l(\mathbf{x})]_{(x,y,c)}$ with c the colour channels.

6.1.2.2 *Optimising B-cos Networks for Classification*

In the following, we discuss why the linear transformations $\mathbf{W}_{1\to L}$ (Equation (6.14)) can be expected to align with relevant input patterns for models based on B-cos transformations.

For this, first note that the output of each neuron—and thus of each layer—is bounded, cf. Equations (6.5) and (6.10). Since the output of a B-cos Network is computed as a sequence of such bounded transformations, see Equation (6.13), the output of the network as a whole is also bounded. Secondly, note that a B-cos Network only achieves its upper bound for a given input if every unit achieves its upper bound. Importantly, the individual units can only achieve their maxima by aligning with their inputs (cf. Equation (6.5)). Hence, optimising a B-cos Network to maximise its output over a set of inputs will optimise the model weights to align with those inputs.

To take advantage of this for B-cos classification models, we train them with the binary cross entropy (BCE) loss

$$\mathcal{L}(\mathbf{x}_i, \mathbf{y}_i) = \text{BCE}\left(\sigma(\mathbf{f}^*(\mathbf{x}_i;\theta)/T + \mathbf{b}), \mathbf{y}_i\right) \tag{6.17}$$

for input $\mathbf{x}_i$ and the respective one-hot encoding of the label $\mathbf{y}_i$; see below (target encoding) for an additional discussion. Here, σ denotes the sigmoid function, $\mathbf{b}$ and T fixed bias and scaling terms, and θ the model parameters. Note that we choose the BCE loss as it directly entails maximisation. Specifically, in order to reduce the BCE loss, the network is optimised to maximise the (negative) class logit for the correct (incorrect) classes. As discussed in the previous paragraph, this will optimise the weights in each layer of the network to align with their inputs. In particular, they will need to align with class-specific input patterns such that these result in large outputs for the respective class logits.

Finally, note that increasing B allows to specifically reduce the output of badly aligned weights in each layer (i.e., with low cosine similarity to the input). This decreases the layer's output strength and thus the output of the network as a whole, which increases the alignment pressure during optimisation (thus, higher B→higher alignment, see Figure 6.4).

Target Encoding and Logit Bias $\mathbf{b}$. As discussed, BCE optimises the models to maximise the absolute magnitude of both the positive (target class) as well as the negative logits (other classes). To encourage the models to find *positive* evidence (i.e. contributions as defined in Equation (6.16)) for the target class, rather than negative evidence for the other classes, we set the logit bias $\mathbf{b}$ to $\log\left(1/(C-1)\right)$ for C classes, which corresponds to a default class probability of $1/C$ for each class. As a result, the magnitude of $\mathbf{f}^*(\mathbf{x}_i;\theta)$ can be small for non-target classes without incurring a large loss, while it has to be large for the target class to achieve a class confidence close to 1 and thus reduce the BCE loss.

Interestingly, in our experiments we found that models with normalisation layers (see Section 6.1.3.2) can still exhibit a tendency to focus on negative evidence, especially when the number of classes C is large (e.g. on ImageNet), see Section 6.3.4. To overcome this, for our ImageNet experiments we additionally modify the target encoding $\mathbf{y}_i$ to be $1/C$ for all non-target classes, which encourages the models to produce weights that are orthogonal to the input for non-target classes, i.e. the optimum can only be reached if $[\mathbf{f}^*(\mathbf{x}_i;\theta)]_c = 0$ for all c that are not the target class.

6.1.3 Advanced B-cos Networks

To test the generality of our approach, we investigate how integrating B-cos transformations into conventional DNNs affects their classification performance and interpretability. To do this, in the following we discuss how to combine B-cos layers and their explanations with common model components, such as activation functions (Section 6.1.3.1), normalisation (Section 6.1.3.2), and attention layers (Section 6.1.3.3).

Note that whenever converting a linear / convolutional layer in a given network to the corresponding B-cos layer, **we remove all bias terms** to ensure that the model output is given by a dynamic linear transformation as in Equation (6.14). For a discussion on the impact of biases, see Section 6.3.4.

6.1.3.1 *Activation Functions in B-cos Networks*

Since B-cos transformations themselves are non-linear, B-cos DNNs do not require activation functions to model non-linear relationships (see Section 6.1.2). Hence, most of our models in Section 6.3 do not employ any activation functions. Nonetheless, combining B-cos Networks with activation functions can still be beneficial. In particular, as we discuss in Section 6.3.1, adding an additional non-linearity allows us to study the effect of the parameter B in Equation (6.3) over a wide range of values, including $\mathrm{B}=1$, and thus to directly compare them to conventional piece-wise linear models[1].

While there are many potential non-linearities to choose from, in this work, we specifically explore the option of combining the B-cos transformation with MaxOut [GWFM+13]. For this, we model every neuron by 2 B-cos transformations of which only the maximum is forwarded to the next layer:

$$\text{mB-cos}(\mathbf{x}) = \max_{i\in\{1,2\}} \{\text{B-cos}(\mathbf{x}; \mathbf{w}_i)\} \ . \tag{6.18}$$

We do so for several reasons. First, this operation maintains the models' dynamic linearity and is thus compatible with the B-cos explanations. Secondly, as discussed above, for $\mathrm{B}=1$ the resulting model is a piece-wise linear model, which allows us to interpolate between conventional piece-wise linear models and the proposed B-cos models. Lastly, while the ReLU [NH10] operation also fulfills these properties, we noticed that B-cos models without any normalisation layers were much easier to optimise with MaxOut instead of ReLU, which we attribute to avoiding 'dying neurons' [GWFM+13].

6.1.3.2 *Adapting Normalisation Layers for B-cos Networks*

In this section we discuss how normalisation layers can be integrated into B-cos Networks. Specifically, we note that these layers (e.g. BatchNorm [IS15], LayerNorm [BKH16], PositionNorm [LWWB19], or InstanceNorm [UVL16]) are (1) non-linear operations and (2) can pose a challenge to the *completeness* of the explanations. By interpreting them as dynamic linear functions and removing bias terms, we can nonetheless seamlessly integrate them into B-cos explanations. First, however, let us define the normalisation layers.

Definition. Normalisation layers have been shown to benefit DNN training by whitening the input $\mathbf{x}$ via

$$*\text{Norm}(\mathbf{x}, \boldsymbol{\mathcal{X}}; \gamma, \beta) = \frac{\mathbf{x} - \langle \boldsymbol{\mathcal{X}} \rangle_*}{\sqrt{\text{var}_*(\boldsymbol{\mathcal{X}})}} \times \gamma + \beta \ , \tag{6.19}$$

with $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{b\times c\times h\times w}$ a batch of b inputs, each with c channels and a width w and a height h; further, $\mathbf{x}\in\mathbb{R}^c$ is a single representation vector from a specific input at a given location. As indicated by the placeholder $*$, the individual normalisation layers differ mainly in the choice of dimensions over which the mean $\langle\cdot\rangle$ and the variance var are computed. Finally, note that at inference time, the mean and variance terms are sometimes replaced by running estimates of those values, most commonly done so in BatchNorm [IS15].

To avoid ambiguities[2] and facilitate the comparison of different normalisation schemes, we define the

[1] For $\mathrm{B}=1$ linear and B-cos transformations are equivalent (Equation (6.4)).

[2] Note that LayerNorm has in fact been interpreted differently by different authors (e.g. [LMW+22] vs. [LWWB19, WH18]), i.e. either as Equation (6.21) or Equation (6.23).

normalisation layers by the respective choice of dimensions they use:

$$\text{BatchNorm}(\mathbf{x}, \boldsymbol{\mathcal{X}}; \gamma, \beta) \coloneqq \text{BHW-Norm}(\mathbf{x}, \boldsymbol{\mathcal{X}}; \gamma, \beta) \tag{6.20}$$

$$\text{LayerNorm}(\mathbf{x}, \boldsymbol{\mathcal{X}}; \gamma, \beta) \coloneqq \text{CHW-Norm}(\mathbf{x}, \boldsymbol{\mathcal{X}}; \gamma, \beta) \tag{6.21}$$

$$\text{InstanceNorm}(\mathbf{x}, \boldsymbol{\mathcal{X}}; \gamma, \beta) \coloneqq \text{HW-Norm}(\mathbf{x}, \boldsymbol{\mathcal{X}}; \gamma, \beta) \tag{6.22}$$

$$\text{PositionNorm}(\mathbf{x}, \boldsymbol{\mathcal{X}}; \gamma, \beta) \coloneqq \text{C -Norm}(\mathbf{x}, \boldsymbol{\mathcal{X}}; \gamma, \beta) \tag{6.23}$$

i.e. BatchNorm estimates mean and variance for each channel independently (no 'C') over all spatial dimensions (HW) and inputs in the batch (B), whereas LayerNorm computes mean and variance across all activations (CHW) in a single input, but does not use other inputs to estimate those values.

Using the above definitions, an additional candidate naturally emerges, which we call *AllNorm*:

$$\text{AllNorm}(\mathbf{x}, \boldsymbol{\mathcal{X}}; \gamma, \beta) \coloneqq \text{BCHW-Norm}(\mathbf{x}, \boldsymbol{\mathcal{X}}; \gamma, \beta)\,. \tag{6.24}$$

Dynamic Linear Normalisation. Normalising the input by the variance of course constitutes a highly non-linear operation. To nonetheless integrate this operation into the B-cos-based dynamic linear explanations (cf. Equation (6.13)), we interpret the normalisation itself to be a dynamic linear function with:

$$\mathbf{w}_{*\text{Norm}}(\mathbf{x}, \boldsymbol{\mathcal{X}}) = \gamma \times \sqrt{\text{var}_*^{-1}(\boldsymbol{\mathcal{X}})} \tag{6.25}$$

$$\text{s.t. } *\,\text{Norm}(\mathbf{x}, \boldsymbol{\mathcal{X}}; \gamma, \beta) = \mathbf{w}_{*\text{Norm}} \times (\mathbf{x} - \langle \boldsymbol{\mathcal{X}} \rangle_*) + \beta\,, \tag{6.26}$$

Explanation Completeness. As becomes clear from Equation (6.19), normalisation layers introduce additive biases into the model prediction (i.e. β and, if running estimates are used, by the running mean estimation of $\langle \boldsymbol{\mathcal{X}} \rangle_*$). The model output is thus not given by a dynamic linear transformation anymore (cf. Equation (6.14)), but instead by an *affine* transformation $\mathbf{f}^*(\mathbf{x}; \theta) = \mathbf{W}(\mathbf{x})\mathbf{x} + \mathbf{b}(\mathbf{x})$ with an input-dependent bias $\mathbf{b}(\mathbf{x})$.

The contribution maps in Equation (6.16) would thus not be *complete* (i.e. the sum over contributions would not yield the respective class logit), as they do not take the biases into account. This, however, is undesirable for model-faithful explanations [SF19, STY17], as the bias terms can play a significant role in the model decision (see Section 6.3.4).

To alleviate this, we use bias-free (BF) normalisation [MKSFG20], and remove the *learnt* biases β, which would make a linear summary of the model via $\mathbf{W}(\mathbf{x})\mathbf{x}$ incomplete:

$$*\text{Norm}^{\text{BF}}(\mathbf{x}, \boldsymbol{\mathcal{X}}; \gamma, \beta) = \frac{\mathbf{x} - \langle \boldsymbol{\mathcal{X}} \rangle_*}{\sqrt{\text{var}_*(\boldsymbol{\mathcal{X}})}} \times \gamma + 0\,. \tag{6.27}$$

Moreover, for normalisation layers that replace the mean computation $\langle \boldsymbol{\mathcal{X}} \rangle_*$ by a running mean estimate[3] at inference time (i.e. BatchNorm), we additionally remove the centring term $\langle \boldsymbol{\mathcal{X}} \rangle_*$, since otherwise external biases would be introduced at inference time.

6.1.3.3 *B-cos ViTs: Combining B-cos and Attention Layers*

Thus far, our discussion focused on designing interpretable convolutional neural networks (CNNs), which have long dominated computer vision research. Recently, however, CNNs are often surpassed by

[3] Note that the subtraction of the mean $\langle \boldsymbol{\mathcal{X}} \rangle_*$ does not introduce *external* biases that would be unaccounted for in the explanations, as this operation can be modelled by a linear transformation of the input.

transformers [VSP+17], which—if the current development is any indication—will replace CNNs for ever more tasks and domains. In the following, we therefore investigate how to design B-cos transformers.

For this, first note that the core difference between vision transformers (ViTs) and CNNs is the use of attention layers and additive positional encodings. The remaining ViT components (tokenisation, MLP blocks, normalisation layers, classification head) have a direct convolutional counterpart.

In particular, the tokenisation module is typically given by a $K \times K$ convolution with a stride and kernel size of K, the MLP blocks effectively perform 1x1 convolutions (cf. [LMW+22]), the normalisation layers can be expressed as PositionNorm in CNNs (Equation (6.23)) and the classification head is either given by a single or multiple linear layers. Therefore, in the following, we specifically discuss the attention mechanism and the positional embeddings in more detail.

Attention. Interestingly, the attention operation itself is already dynamic linear and attention thus lends itself well to be integrated into the linear summary according to Equation (6.14):

$$\begin{aligned} \text{Att}(\mathbf{X}; \mathbf{Q}, \mathbf{K}, \mathbf{V}) &= \underbrace{\text{softmax}\left(\mathbf{X}^T\mathbf{Q}^T\mathbf{K}\mathbf{X}\right)}_{\text{Attention matrix } \mathbf{A}(\mathbf{X})} \underbrace{\mathbf{V}\mathbf{X}}_{\text{Value}(\mathbf{X})} && (6.28) \\ &= \underbrace{\mathbf{A}(\mathbf{X})\,\mathbf{V}}_{\mathbf{W}(\mathbf{X})}\,\mathbf{X} \;=\; \mathbf{W}(\mathbf{X})\mathbf{X}. && (6.29) \end{aligned}$$

Here, $\mathbf{Q}, \mathbf{K}$ and $\mathbf{V}$ are the query, key, and value transformation matrices and $\mathbf{X}$ denotes the matrix of input tokens to the attention layer. Softmax is computed column-wise.

In multi-head self-attention (MSA), H attention heads are used in parallel. Their concatenated outputs are then linearly projected by a matrix $\mathbf{U}$ via

$$\text{MSA}(\mathbf{X}) = \mathbf{U}\left[\mathbf{W}_1(\mathbf{X})\mathbf{X},\ \ldots\ , \mathbf{W}_H(\mathbf{X})\mathbf{X}\right]\ , \tag{6.30}$$

which is still dynamic linear. While there are multiple linear operations involved in MSA (query, key, value, and projection computations) and thus various options for introducing B-cos layers, we empirically found not replacing[4] the query, key, and value computations to yield the best results, which we attribute to a potentially higher compatibility with softmax. Hence, to adapt the attention layers to the B-cos framework, we only replace the linear projection via $\mathbf{U}$ by a B-cos transformation in our experiments.

Positional Encoding. In contrast to CNNs, which possess a strong inductive bias regarding spatial relations (local connectivity), transformers are invariant with respect to the token order and thus lack such a 'locality bias'. To nevertheless leverage spatial information, it is common practice to break the symmetry between tokens by adding a (learnt) embedding $\mathbf{E}$ to the input tokens $\mathbf{X}$ such that $\mathbf{X}' = \mathbf{X} + \mathbf{E}$.

While explanations solely with respect to $\mathbf{X}$ are not thus complete (cf. Section 6.1.3.2), since part of the model output will be based on contributions from $\mathbf{E}$, we experimentally find that the positional embeddings do not negatively impact the interpretability of the ViTs. How to design ViT models without the need for such external biases represents an interesting direction for future work.

6.2 EXPERIMENTAL SETTING

Datasets. We evaluate the accuracies of a wide range of B-cos Networks on the CIFAR10 [Kri09] and the ImageNet [DDS+09] datasets. We use the same datasets for the qualitative and quantitative evaluations of the model-inherent explanations.

[4] I.e., these transformations effectively use a B-cos layer with $B = 1$.

CIFAR10 Models. For the CIFAR10 experiments, we develop a simple fully-convolutional B-cos DNN, consisting of 9 convolutions, each with a kernel size of 3, followed by a global pooling operation. We evaluate a network without additional non-linearities as well as with MaxOut units, see Section 6.1.3.1. Additionally, to study the normalisation layers and the effect of the bias terms (cf. Equation (6.27)), we evaluate various B-cos ResNets. For this we adapt a conventional ResNet-56 [HZRS16] by removing all activation functions and bias parameters and replacing all linear / convolutional layers by their corresponding B-cos version. Additionally, we replace the conventional batch normalisation layers by the (modified) normalisation layers as described in Section 6.1.3.2.

ImageNet Models. For the ImageNet experiments, we rely on the publicly available [PGM+19] implementations of a wide range of CNNs (VGGs [SZ15], ResNets [HZRS16], DenseNets [HLVDMW17], ResNext [XGD+17] and ConvNext models [LMW+22]). Further, we evaluate B-cos versions of the Vision Transformers (ViTs) [DBK+21], specifically the version from [BZK22], as well as ViT$_C$ models as in [XSM+21], i.e. ViTs with a shallow stem of four convolutional layers. We adapt all of those architectures to B-cos Networks as described in Section 6.1.3 and train them according to standard protocols. Specifically, we train most CNNs with the default 90 epochs training paradigm as found in the torchvision library [Tor]; additionally, we evaluate various models when employing a more sophisticated training recipe [Tor21] of 600 epochs, based on the ConvNext [LMW+22] training protocol. Finally, for the ViT models, we follow the protocol proposed by [BZK22], which has shown strong performance for ViTs with only 90 epochs of training. For more details on the training procedure, see Appendix Section C.3

Image Encoding. We add three additional channels and encode images as $[r, g, b, 1-r, 1-g, 1-b]$, with $r, g, b \in [0, 1]$ the red, green, and blue colour channels. On the one hand, this reduces a bias towards bright regions in the image[5] [BFS21]. On the other hand, colors with the *same angle in the original encoding*—i.e., $[r_1, g_1, b_1] \propto [r_2, g_2, b_2]$—*are unambiguously encoded by their angles under the new encoding*. Therefore, the linear transformation $\mathbf{W}_{1\to l}$ can be decoded into colors just based on the angles of each pixel, see Figure 6.1. For a detailed discussion, see Appendix Section C.4.3.

Evaluating Explanations. To compare explanations for the model decisions and evaluate their faithfulness, we employ the *grid pointing game* [BFS21]. I.e., we evaluate the trained models on a synthetic 3x3 grid of images of different classes and for each of the corresponding class logits measure how much positive attribution an explanation method assigns to the correct location in the grid, see Figure 6.3 for an example. Note that due to the global attention in ViTs such grids are much further out of distribution at every layer of the model than they are for CNNs, for which the locally applied convolutional kernels are not affected by the synthetic nature of the input grids. The grid pointing game, however, relies on the assumption that the model extracts similar features in the synthetic setting and *locally* correctly classifies the subimages. To ensure that this assumption (at least approximately) holds, we apply the ViTs in a sliding window fashion to the synthetic image grids, i.e. we apply the ViTs to patches of size 224x224 extracted at a stride of 112 from the grid. As this significantly increases the computational cost, we use 2x2 grids for ViTs.

Following [BFS21], we construct 500 grids from the most confidently and correctly classified images and compare the model-inherent contribution maps (Equation (6.16)) against several commonly employed post-hoc explanation methods under two settings. First, we evaluate all methods on B-cos Networks to investigate which method provides the best explanation *for the same model*. Secondly, we further evaluate the post-hoc methods on pre-trained versions of the original models. This allows to compare explanations *between different models* and assess the *explainability gain* obtained by converting conventional models to B-cos Networks. Lastly, all attribution maps (except for GradCAM, which is of low resolution to begin

[5]Note that the model output is given by a weighted sum over the input (Equation (6.14)). Since black pixels are encoded as zero in the conventional encoding, they cannot contribute to the class logits (cf. Equation (6.16)).

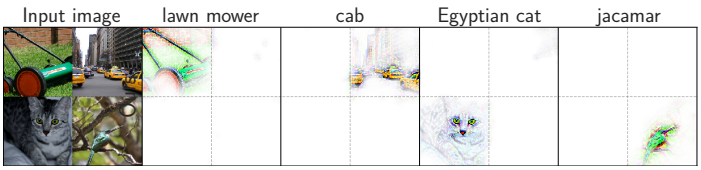


Figure 6.3: **2×2 pointing game example.** *Col. 1*: input image. *Cols. 2–5*: explanations for individual class logits.

with) are smoothed by a 15×15 (3×3) kernel to better account for negative attributions in the localisation metric for ImageNet (CIFAR10) images, which is small w.r.t. the image size of 224×224 (32×32).

Visualisations Details. For generating the visualisations of the linear transforms for individual neurons n in layer l (cf. Figures 6.1 and 6.10), we proceed as follows. First, we select all pixel locations (x, y) that positively contribute to the respective activation (e.g., class logit) as computed by Equation (6.16); i.e., $\{(x, y)$ s.t. $\sum_c [\mathbf{s}_n^l(\mathbf{x})]_{(x,y,c)} > 0\}$ with c the 6 colour channels (see image encoding). Then, we normalise the weights of each colour channel such that the corresponding weights (e.g., for r and $1-r$) sum to 1. Note that this normalisation maintains the angle for each colour channel pair (i.e., r and $1-r$), but produces values in the allowed range $r, g, b \in [0, 1]$. These normalised weights can then directly be visualised as colour images. The opacity of a pixel is set to $\min(\|\mathbf{w}_{(x,y)}\|_2 / p_{99.9}, 1)$, with $p_{99.9}$ the 99.9th percentile over the weight norms $\|\mathbf{w}_{(x,y)}\|_2$ across all (x, y).

6.3 RESULTS

In this section, we analyse the performance and interpretability of B-cos models. For this, in Section 6.3.1 we show results of 'simple' B-cos models without advanced architectural elements such as skip connections. In this context, we investigate how the B parameter influences B-cos models in terms of performance and interpretability. Thereafter, in Section 6.3.2, we present quantitative results of the *advanced* B-cos models, i.e., B-cos models based on common DNN architectures (cf. Section 6.1.3). In Section 6.3.3, we visualise and qualitatively discuss explanations for individual neurons of the advanced B-cos models, and finally, in Section 6.3.4, we discuss the impact of the bias terms in the normalisation layers.

6.3.1 Simple B-cos Models

In the following, we discuss the results of simple B-cos models evaluated on the CIFAR10 dataset.

	MaxOut B-cos Networks							**plain**
B	1.00	1.25	1.50	1.75	2.00	2.25	2.50	2.00
Accuracy (%)	93.5	93.8	93.7	93.7	93.2	92.6	92.4	91.5

Table 6.1: **CIFAR10 accuracies** of a B-cos model without additional non-linearities (plain) and for B-cos models with MaxOut (Equation (6.18)) and increasing values for B (left to right).

Accuracy. In Table 6.1, we present the test accuracies of various B-cos models trained on CIFAR10. We show that a plain B-cos model (B=2) without any add-ons (ReLU, batch norm, etc.) can achieve competitive[6] performance. By modelling each neuron via 2 MaxOut units (Equation (6.18)), the performance can be increased and the resulting model (B=2) performs on par with a ResNet-56 (achieving 93.0%, see

[6] A ResNet-20 achieves 91.2% [HZRS16] with the same data augmentation.

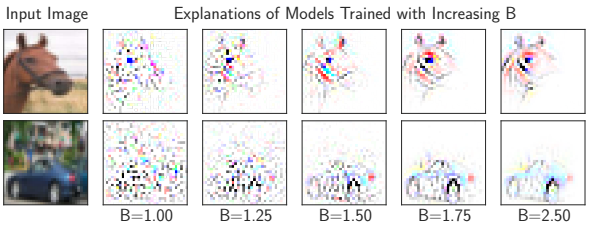


Figure 6.4: **Impact of B.** *Col. 1*: Input images. *Cols. 2–6*: Explanations for different classes c (top: 'horse'; bottom: 'car') of models trained with increasing B. For higher B, the model-inherent linear explanations $[\mathbf{W}_{1\to l}]_c$ increasingly align with discriminative input patterns, thus becoming more interpretable.

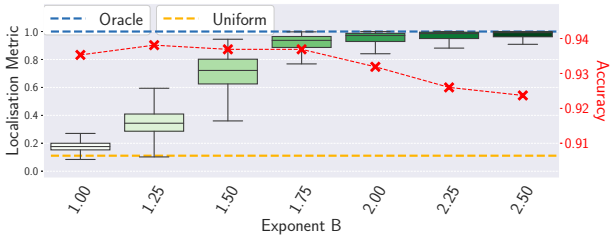


Figure 6.5: **CIFAR10 results.** Accuracy and localisation (crosses / box plots) for a B-cos Network trained with different B. While decreasing accuracy, larger B significantly improve localisation.

[HZRS16]). Further, we see that an increase in the parameter B leads to a decline in performance from 93.8% for $\mathrm{B}=1.25$ to 92.4% for $\mathrm{B}=2.5$. Notably, despite its simple design, our strongest model with $\mathrm{B}=1.25$ performs similarly to the strongest ResNet model (93.6%) reported in [HZRS16].

Model Interpretability. As discussed in Section 6.1.2.2, we expect an increase in B to increase the alignment pressure on the weights during optimisation and thus influence the models' optima, similar to the single unit case in Figure 6.2. This is indeed what we observe. For example, in Figure 6.4, we visualise $[\mathbf{W}_{1\to l}(\mathbf{x}_i)]_{y_i}$ (see Equation (6.14)) for different samples i from the CIFAR10 test set. For higher values of B, the weight alignment increases notably from piece-wise linear models (B=1) to B-cos models with higher B (B=2.5). Importantly, this does not only lead to an increase in the visual quality of the explanations, but also to quantifiable gains in model interpretability. In particular, as we show in Figure 6.5, the spatial contribution maps defined by $\mathbf{W}_{1\to l}(\mathbf{x}_i)$ (see Equation (6.16)) of models with larger B values score significantly higher in the localisation metric (see Section 6.2).

6.3.2 Advanced B-cos Models

B-cos CNNs – Accuracy. In Tables 6.2 and 6.3, we present the top-1 accuracies of the B-cos models on the ImageNet validation set for the 90 epoch and 600 epoch training paradigms respectively. For comparison, we also show the difference to the accuracy of the respective baseline models as reported in [Tor] (Δ^1). As can be seen, the B-cos models are highly competitive, achieving accuracies on par with the baselines for some models (ResNet-50, $\Delta^1 = -0.2$), and a worst-case drop of $\Delta^1 = -2.0$ (ResNext-50-32x4d) under the 90 epoch training paradigm (Table 6.2). Interestingly, we find that the differences to the baseline models vanish when training the baselines without bias terms in the convolution and normalisation layers, i.e. as is done in B-cos models; we denote the difference to the baselines without biases by Δ^2.

When employing a long training protocol with additional data augmentation as in [LMW+22], the performance of the B-cos models can be further improved: e.g., the B-cos ResNet-152 increases its top-1 accuracy by 3.2pp (compare Tables 6.2 and 6.3) and achieves a top-1 accuracy of 80.2%, which is an unprecedented performance for models that inherently provide such detailed explanations (see Section 6.3.3). Nonetheless, especially for the ConvNext models, we observe a significant performance drop with respect to the baseline numbers reported in [Tor] (up to 5.0pp for ConvNext-Tiny). Note, however, that the training protocol has been meticulously optimised for ConvNext models in [LMW+22];

as such, we expect that the performance of B-cos models will significantly improve if a similar effort as in [LMW+22] is undertaken for optimising the model architecture and training procedure.

	Accuracy			Localisation		
B-cos Model	%	Δ^1	Δ^2	%	$\Delta^{\text{B-cos}}_{\text{IntG}}$	$\Delta^{\text{basel.}}_{\text{IntG}}$
VGG-11	69.0	-1.4	-0.6	85.6	+26.4	+64.5
ResNet-18	68.7	-1.1	+0.3	86.9	+34.0	+64.0
ResNet-34	72.1	-1.2	-0.1	89.0	+36.9	+65.2
ResNet-50	75.9	-0.2	+2.4	90.2	+32.7	+63.3
ResNet-101	76.3	-1.1	+6.2	91.8	+33.9	+64.4
ResNet-152	76.6	-1.7	+3.2	91.3	+33.6	+63.9
DenseNet-121	73.6	-0.8	+0.2	92.1	+41.2	+69.1
DenseNet-161	76.6	-0.5	+5.2	93.4	+47.1	+68.5
DenseNet-169	75.0	-0.6	+0.3	91.8	+49.3	+67.7
DenseNet-201	75.6	-1.3	+0.3	93.0	+44.8	+68.8
ResNext-50-32x4d	75.6	-2.0	+0.6	91.2	+28.2	+64.1

Table 6.2: **Left:** Top-1 accuracies on ImageNet for a standard 90 epoch training protocol [Tor]. Δ^1: Accuracy difference with respect to the numbers reported in [Tor]. Δ^2: Accuracy difference with respect to baselines trained without biases. This setting puts the baselines and the B-cos models on an equal footing, as B-cos models do not use biases either. **Right:** Additionally, we show localisation scores of the model-inherent explanations (Equation (6.16)) as well as localisation scores of IntGrad evaluated on the B-cos models ($\Delta^{\text{B-cos}}_{\text{IntG}}$) and the pre-trained baselines ($\Delta^{\text{basel.}}_{\text{IntG}}$) as obtained from [Tor]. We observe significant localisation gains for all models.

	Accuracy		Localisation		
B-cos Model	%	Δ^1	%	$\Delta^{\text{B-cos}}_{\text{IntG}}$	$\Delta^{\text{basel.}}_{\text{IntG}}$
ResNet-50	79.5	-1.4	86.2	+41.0	+58.6
ResNet-152	80.2	-2.1	85.3	+44.2	+55.5
ConvNext-Tiny	77.5	-5.0	69.6	+37.7	+46.5
ConvNext-Base	79.7	-4.4	81.4	+35.2	+54.6

Table 6.3: **Left:** ImageNet accuracies for the ConvNext-inspired training protocol [Tor21]. Δ^1: Accuracy difference with respect to the numbers reported in [Tor21]. While we observe a significant drop with respect to the ConvNext models, we note that the training protocol has been optimised explicitly for those architectures [LMW+22] and that B-cos models do not employ biases (cf. Table 6.2) to ensure $\mathbf{y}(\mathbf{x}) = \mathbf{W}(\mathbf{x})\mathbf{x}$. **Right:** Additionally, we show the localisation scores of the model-inherent explanations (Equation (6.16)) as well as the localisation scores of IntGrad evaluated on the B-cos models ($\Delta^{\text{B-cos}}_{\text{IntG}}$) and the pre-trained baselines ($\Delta^{\text{basel.}}_{\text{IntG}}$) as obtained from [Tor]. We observe significant localisation gains for all models.

B-cos CNNs – Interpretability. In addition to the model accuracy, in Tables 6.2 and 6.3 we provide the mean localisation scores obtained in the grid pointing game (cf. Figure 6.3) of the model-inherent explanations according to Equation (6.16) as well as the difference to one of the most popular post-hoc attribution methods (IntGrad) evaluated on the B-cos models ($\Delta^{\text{B-cos}}_{\text{IntG}}$) and the baselines ($\Delta^{\text{basel.}}_{\text{IntG}}$).

Across all models, the model-inherent explanations achieve significantly better localisation than IntGrad; this difference is even more pronounced when comparing to IntGrad explanations for the baselines.

To provide more detail, in Figure 6.6 we show the localisation results for two specific models (ResNet-152, DenseNet-201) for a wide range of post-hoc explanation methods: the 'vanilla' gradient (Grad) [BSH+10],

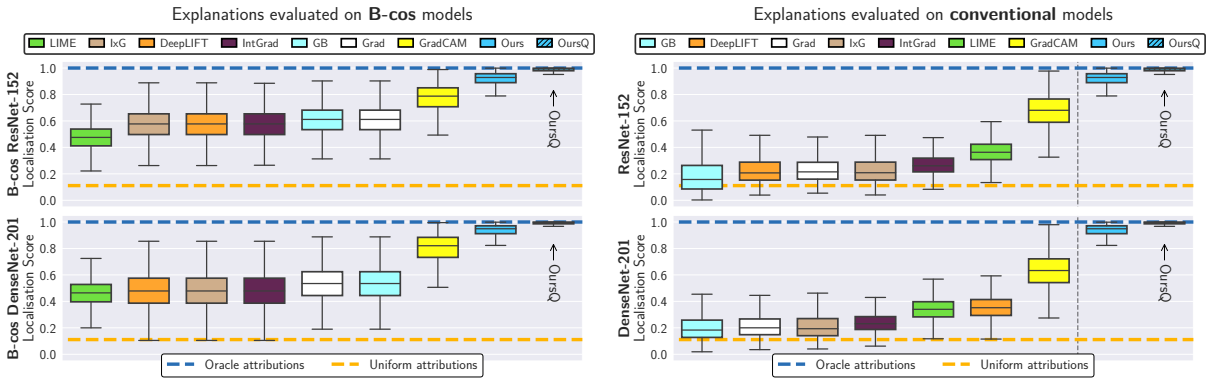


Figure 6.6: **Localisation results** of the model-inherent contribution maps ('Ours', Equation (6.16)) and various post-hoc explanation methods for a B-cos ResNet-152 and a B-cos DenseNet-201 (*left*) and their conventional counterparts (*right*); we further show the results of the top 0.025 percentile of pixels in the model-inherent explanations ('OursQ'), i.e. roughly 11.3k pixels, cf. Figure 6.7. For an easier cross-model comparison, we repeat the results of the B-cos models in right column (Ours + OursQ). The model-inherent explanations significantly outperform post-hoc explanation methods when evaluating those methods on the B-cos models and on the conventional counterparts.

Guided Backpropagation (GB) [SDBR15], Input $\times$ Gradient (IxG) [SGK17], Integrated Gradients (IntGrad) [STY17], DeepLIFT [SGK17], GradCAM [SCD+17], and LIME [RSG16]. In particular, we plot the localisation scores for the B-cos versions of those models (left) and the respective baselines (right).

In comparison to the post-hoc explanation methods, we observe the model-inherent explanations ('Ours', Equation (6.16)) to consistently and significantly outperform all other methods, both when comparing to post-hoc explanations for the *same* model (left), as well as when comparing explanations *across* models (for ease of comparison, we repeat the B-cos results on the right). Note that while GradCAM also shows strong performance, it explains only a fraction of the entire model [RBS22] and thus yields much coarser attribution maps, see also Figure 6.9. Interestingly, we also find that the other post-hoc explanations consistently perform better for B-cos models than for the baseline models (compare row-wise).

Finally, we additionally evaluate how well the most highly contributing pixels according to $\mathbf{W}(\mathbf{x})$ fare in the localisation metric, which we denote as OursQuantile (OursQ) in Figure 6.6, see also Figure 6.7. We find that when using only the top-n contributing pixels in the model-inherent explanations, the localisation score can be improved even further.

B-cos ViTs – Accuracy. In Table 6.4, we report the top-1 accuracies of B-cos ViT and ViT$_C$ models of various commonly used sizes—Tiny (Ti), Small (S), Base (B), and Large (L)—and the difference (Δ^3) to the respective baseline models which we optimised under the same training protocol [BZK22].

For both the original ViTs and the ones with a shallow convolutional stem (ViT$_C$), we observe significant performance drops for smaller model sizes (Ti+S). For larger models (B+L), however, the difference between the B-cos models and the baselines vanish, see, e.g., the ViT$_C$-B ($\Delta^3 = +0.3$) and the ViT$_C$-L ($\Delta^3 = -0.1$) models. Further, and in line with [XSM+21], we find that a shallow convolutional stem leads to significant performance gains; as we discuss next, it can also significantly improve interpretability.

B-cos ViTs – Interpretability. To assess the B-cos ViTs' interpretability, we compare the model-inherent explanations (Equation (6.16)) to IntGrad evaluated on the B-cos ($\Delta^{\text{B-cos}}_{\text{IntG}}$) and the baseline models ($\Delta^{\text{basel.}}_{\text{IntG}}$); further, we report the difference to a commonly used attention-based explanation, namely

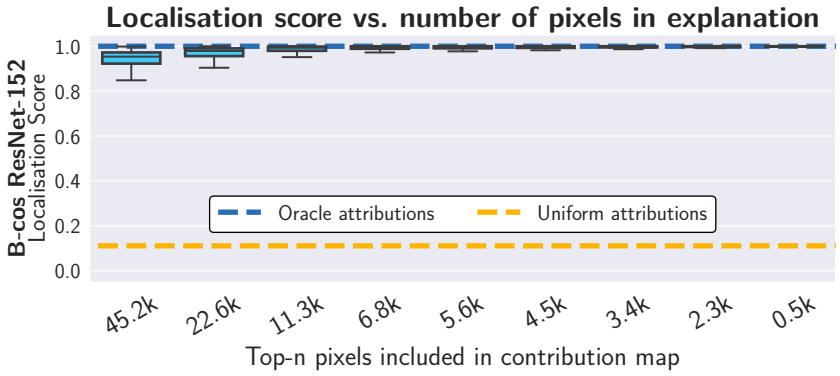


Figure 6.7: **Top-n pixel localisation.** The localisation score significantly improves when only including the n most strongly contributing pixels of the model-inherent explanations in the metric. While highlighting small contributions coming from images belonging to other classes (see Figure 6.3) might be model-faithful, this is penalised in the metric. Nonetheless, the correct region is consistently highlighted by the most strongly contributing pixels.

Attention Rollout [AZ20] ($\Delta^*_{\text{Rollout}}$). Since Rollout is inherently class-agnostic[7], it on average yields the same localisation as uniformly distributed importance values for all models, i.e. a mean score of 25%.

As can be seen in Table 6.4, the B-cos ViT and ViT$_C$ models show significant gains in localisation compared to the respective baseline models. Interestingly, adding a convolutional stem of just four convolutional layers yields notable improvements in localisation between the respective B-cos models, i.e. between B-cos ViT and B-cos ViT$_C$ models. This suggests that ViT$_C$ models perform better because the transformers are applied to more meaningful representations, which avoids mixing visually similar, yet semantically unrelated, patches in the first few global attention operations.

These *quantitative* improvements in interpretability between ViT and ViT$_C$ models can also be observed qualitatively, see Figure 6.8. Specifically, in contrast to the explanations of the ViTs without convolutional stem (row 2), the B-cos ViT$_C$ explanations are visually more coherent and less sparse, making them more easily interpretable for humans.

6.3.3 Qualitative Evaluation of Explanations

The following is based on the DenseNet-121, similar results were observed for other B-cos models.

Every activation in a B-cos model is the result of a sequence of B-cos transforms. Hence, the intermediate activations in any layer l can also be explained via the corresponding linear transform $\mathbf{W}_{1\to l}(\mathbf{x})$, see Equation (6.14). For example, in Figure 6.1, we visualise the linear transforms of the respective *class logits* for various input images. Given the alignment pressure during optimisation, these linear transforms align with class-discriminant patterns in the input and thus actually resemble the class objects.

Similarly, in Figures 6.10 and 6.11, we visualise explanations for *intermediate features*. These features are selected from the most highly contributing feature directions $\hat{\mathbf{a}}^l_{(x,y)} \in \mathbb{R}^{d_l}$ according to the linear transformation $\mathbf{W}_{l\to L}(\mathbf{x})$, cf. Equation (6.14); here, $\hat{\mathbf{a}}$ denotes that $\hat{\mathbf{a}}$ is of unit norm, d_l is the number of features in layer l, and (x, y) the position at which this feature is extracted from the activation map.

Specifically, in Figure 6.11, we show explanations for some images that most strongly activate those feature directions across the validation set. We find that features in early layers seem to represent low-level

[7]Rollout computes the product of the average (across heads) attention matrices of all layers and is thus not class-specific.

	Accuracy		Localisation			
B-cos Model	%	Δ^3	%	$\Delta^*_{\text{Rollout}}$	$\Delta^{\text{B-cos}}_{\text{IntGrad}}$	$\Delta^{\text{basel.}}_{\text{IntGrad}}$
ViT-Ti	60.0	-10.3	69.8	+44.8	+12.2	+30.0
ViT-S	69.2	-5.2	72.1	+47.1	+15.0	+30.9
ViT-B	74.4	-0.9	74.6	+49.6	+19.4	+32.5
ViT-L	75.1	-0.7	74.0	+49.0	+17.7	+32.2
ViT$_C$-Ti	67.3	-5.3	86.8	+61.8	+30.3	+46.4
ViT$_C$-S	74.5	-1.2	87.4	+62.4	+30.4	+45.9
ViT$_C$-B	77.1	+0.3	86.5	+61.5	+28.6	+44.5
ViT$_C$-L	77.8	-0.1	87.3	+62.3	+29.9	+45.3

Table 6.4: **Left:** Top-1 accuracies on ImageNet for the 90 epoch Simple-ViT training protocol [BZK22]. Δ^3: Accuracy difference with respect to training the baseline models according to the same protocol. While we observe a significant drop for smaller ViT and ViT$_C$ models, the difference to the baseline models vanishes for larger ViTs: e.g., the B-cos ViT$_C$-B and ViT$_C$-L models perform on par with the baselines. **Right:** Additionally, we show the localisation scores of the model-inherent explanations (Equation (6.16)) as well as the differences to the scores obtained via Attention Rollout ($\Delta^*_{\text{Rollout}}$) and IntGrad evaluated on the B-cos models ($\Delta^{\text{B-cos}}_{\text{IntG}}$) and the baselines ($\Delta^{\text{basel.}}_{\text{IntG}}$). We observe significant localisation gains when using the B-cos explanations.

concepts, and become more complex in later layers.

Figure 6.10 shows additional results for features in layer 87. We observed that some features are highly specific to certain concepts, such as bike wheels (feature 89), red bird beaks (feature 109), or faces with headgear (feature 123). Importantly, these features do not just learn to align with simple, fixed patterns—instead, they represent semantic concepts and are robust to changes in colour, size, and pose.

Lastly, in Figure 6.12, we show explanations of the two most likely classes for images for which the model produces predictions with high uncertainty; additionally, we show the Δ-Explanation, i.e., the difference in contribution maps for the two classes, see Equation (6.16). By means of the model-inherent linear mappings $\mathbf{W}_{1\to L}$, the model can provide a human-interpretable explanation for its uncertainty: there are indeed features in each of those images that provide evidence for both of the predicted classes.

6.3.4 Explicit and Implicit Model Biases

As discussed in Section 6.1.3.2, we design the B-cos CNNs to be bias-free by fixing the *explicit* bias parameters in all layers to be zero. In this section, we aim to motivate this choice further. Additionally, we show that even when doing so, the models can learn to counteract this and add *implicit* biases by exploiting reliable input features such as image edges.

Explicit Biases. As discussed by [SF19] for piece-wise linear networks, the biases of DNNs are often not accounted for in input-level attributions, despite the fact that they can play a critical role in the prediction. Similarly, the contributions $\mathbf{s}_c$ (Equation (6.16)) from the input to the c-th class logit are not complete explanations if biases are used within B-cos models (cf.Section 6.1.3.2). Instead, the output $\mathbf{y}(\mathbf{x})$ is

$$\mathbf{y}(\mathbf{x}) = \mathbf{W}(\mathbf{x})\mathbf{x} + \mathbf{b}(\mathbf{x}) \quad , \tag{6.31}$$

with $\mathbf{b}(\mathbf{x}) \in \mathbb{R}^c$ subsuming all the contributions to the class logits that are not accounted for by the dynamic linear mapping $\mathbf{W}(\mathbf{x})$. As such, B-cos DNNs face the same problem as described by [SF19]: while explanations based on $\mathbf{W}(\mathbf{x})$ neglect the impact of the contributions from bias terms $\mathbf{b}(\mathbf{x})$, in Figure 6.13 we show that the bias term can play a critical role in the model decision. In particular, we find that for

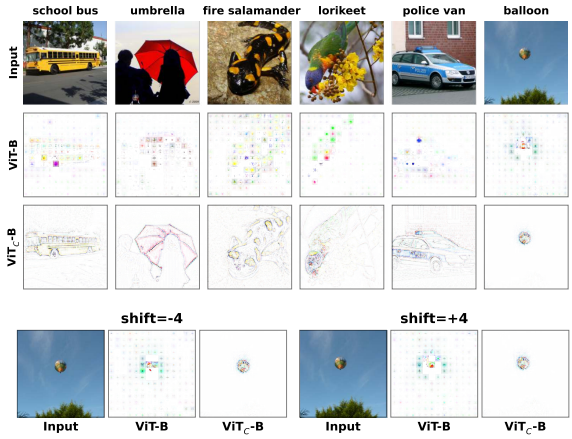


Figure 6.8: **Comparing B-cos ViT and ViT$_C$ models.** *Top:* Explanations of B-cos ViT-B models without (ViT, *row 2*) and with (ViT$_C$, *row 3*) convolutional stem. By adding just 4 convolutional layers for embedding the image patches, the explanations become much more coherent and less sparse. *Bottom:* For a single image, we qualitative visualise how the explanations of the different ViT models perform under diagonal shifts $s \in -4, +4$ of the image on the top right. While both models generally show consistent explanations, the inductive bias of the convolutional stem leads to more stable behaviour and explanations of the ViT$_C$ models.

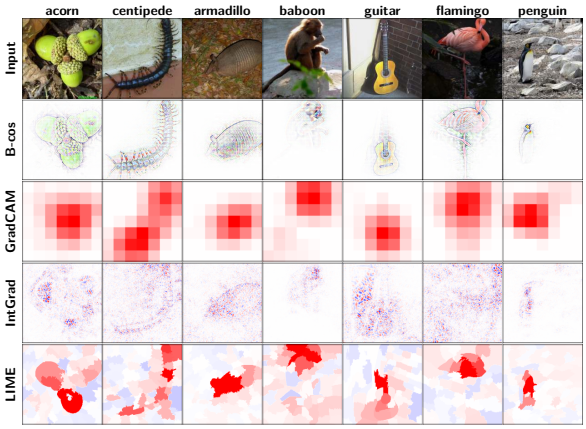


Figure 6.9: **Comparison to post-hoc explanations.** For the same model (a B-cos DenseNet-121), we visualise B-cos explanations as well as several of the most commonly used post-hoc explanation methods (GradCAM, IntGrad, and LIME). For all of the post-hoc explanations, positive, negative, and zero attributions are shown in red, blue, and white respectively. While we find all explanations to generally highlight similar image regions as the B-cos explanations—in particular GradCAM and LIME—the model-inherent linear mappings provide the greatest amount of detail and highlight even fine-grained details such as the feet of the centipede.

some models (e.g. see the models trained with InstanceNorm or BatchNorm), the bias terms make up most of the difference between the top-2 predictions of the model. To ensure that the explanations of B-cos CNNs are complete, we set all bias parameters of the model to zero, see Section 6.1.3.2.

Implicit Biases. Even when no *explicit* bias parameters are used, we found that some models learnt to use image edges for computing biases to add to the class logits. While these biases are correctly reflected in the explanations (see Figure 6.14, row 2), this behaviour makes the model explanations less human-interpretable. Interestingly, these models seem to have learnt to solve the optimisation task in Equation (6.17) in an unintended manner. Specifically, all classes receive highly positive contributions from image corners or edges (the top edge in Figure 6.14), thus yielding high class logits for all classes. To still obtain low scores for the non-target classes, the models then seem to use features of the recognised object in the image to add negative contributions to those classes.

To corroborate this, we additionally visualise the explanations for the mean-corrected logits $\mathbf{y} - \langle \mathbf{y}_c \rangle_c$ in row 3 of Figure 6.14; note that given the dynamic linearity of the B-cos models, this is equivalent to computing mean-corrected explanations $\mathbf{W}(\mathbf{x}) - \langle [\mathbf{W}(\mathbf{x})]_c \rangle_c$. As expected, we find that those modified explanations E' exhibit a high degree of specificity and are thus more easily human-interpretable.

We hypothesise that this behaviour is rooted in the formulation of the optimisation task itself: in particular, the BCE loss 'asks' the model to produce highly negative logits for all classes but one, which might bias the model towards using object features to compute negative contributions.

To overcome this, and to incentivise the models to use object features primarily to compute *positive contributions* for the classes this feature belongs to, we propose to change the optimisation task itself, as

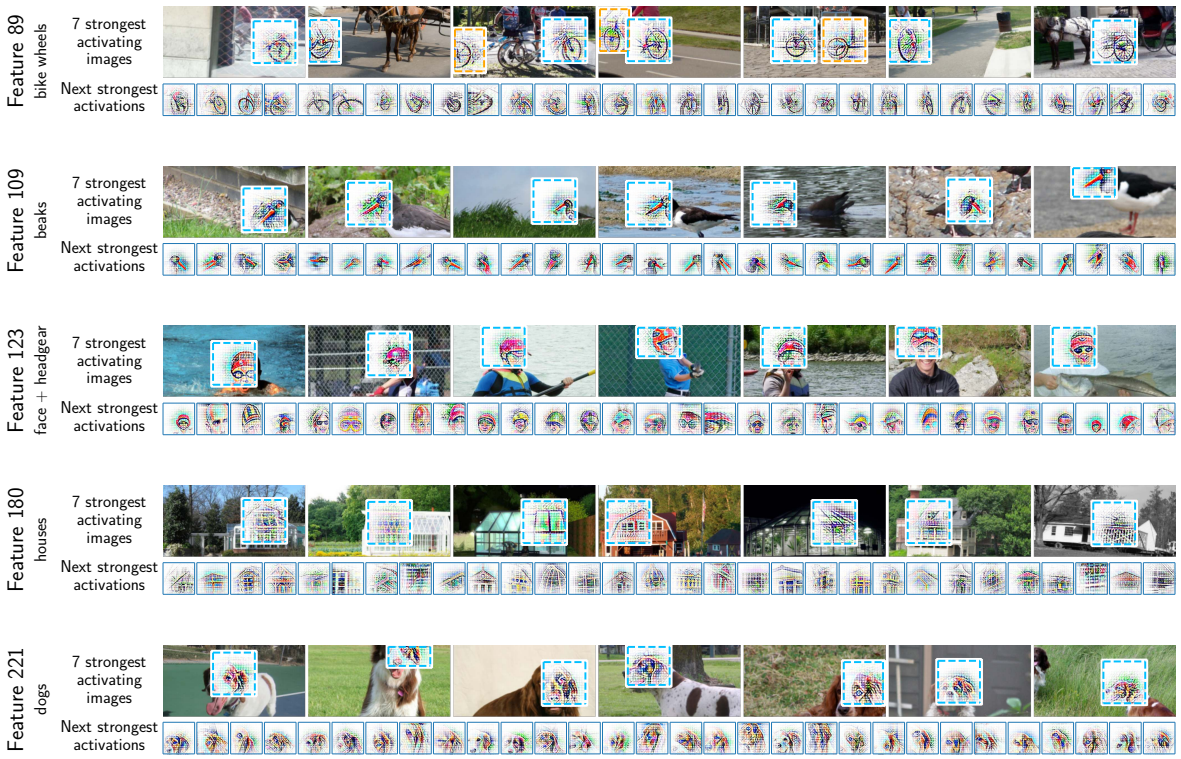


Figure 6.10: **Explanations of 5 highly contributing feature directions in layer 87 of a DenseNet-121**. For each feature, we provide its index number n and a description. Further, we show the 7 most activating images for each feature (top row per feature), in which we visualise the explanation for the highest (*blue squares*) activation; i.e., we visualise the 72×72 centre patch of the weighting $[\mathbf{W}_{1\to l}(\mathbf{x})]_n$ for feature n. For some images in the first row, we further show the explanation for the 2nd highest activation (*orange squares*). Lastly, we show the explanations of the highest activations (corresponding to the blue squares) for the next 30 images to highlight the features' specificity.

described in Section 6.1.2.2. As we show in the last row of Figure 6.14, this indeed has the intended effect and results in explanations that are highly focused on the class objects in the image.

We believe this to highlight the importance of a holistic approach towards interpretable deep neural networks. In particular, these results show that both the model design as well as the optimisation procedure can significantly impact the model behaviour as well as the resulting explanations, which complicates the development of post-hoc explanations methods that do not take those aspects into account.

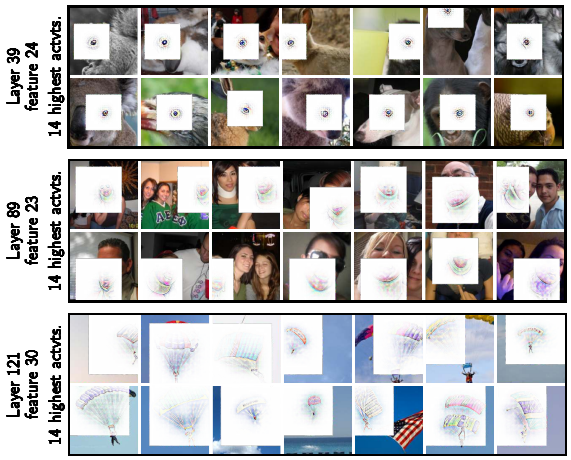


Figure 6.11: **Explanations for highly contributing feature directions in various layers**. In early layers, the features seem to encode low-level concepts (e.g., eyes in layer 39) and represent more high-level concepts in later layers (e.g., layers 89 and 121), such as neck braces or parachutes, see also Figure 6.10.

Figure 6.12: **Ambiguous examples.** *Col. 1:* Input image. *Cols. 2+3:* Explanations for the top-2 predictions of a B-cos DenseNet-121. *Col. 4:* Contribution difference for the two class logits, i.e., $\Delta(c_1, c_2) = \mathbf{s}^L_{c_1}(\mathbf{x}) - \mathbf{s}^L_{c_2}(\mathbf{x})$, see Equation (6.16); positive values shown in orange (c_1), negative ones in blue (c_2).

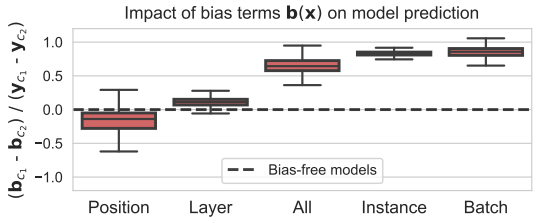


Figure 6.13: **Ratio of bias to logit differences** for the top-2 predictions (c_1, c_2) of ResNet-20 models on the CIFAR10 test set, trained with different normalisation layers. We find that for some of the models, the bias term accounts for most of the difference and thus plays a critical role in the predictions (e.g., for BatchNorm and InstanceNorm). On the other hand, for the B-cos model trained with LayerNorm, the bias term does not seem to be decisive, whereas for PositionNorm it on average even seems to favour the second highest prediction. To avoid not adequately representing the potentially critical bias $\mathbf{b}(\mathbf{x})$ in the explanations, we ensure $\mathbf{b}(\mathbf{x}) = 0$ for B-cos CNNs.

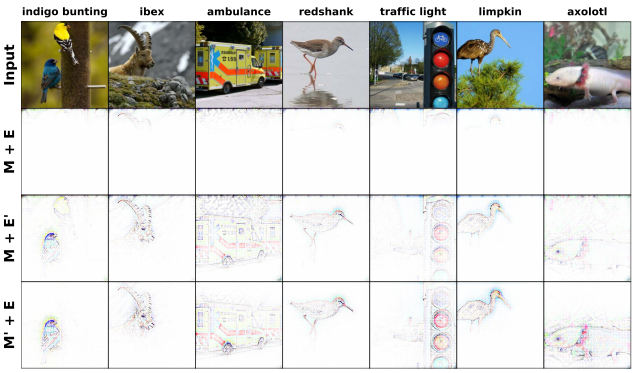


Figure 6.14: **'You get what you optimise for.'** We found that for some models M (here, ResNet-18), the explanations E highlight image edges for all samples, implying that most *positive* contributions for all classes come from those edges, see **row 2**. However, the explanations for the difference from the mean logit (i.e., $\mathbf{y}_c - \langle \mathbf{y}_j \rangle_j$)—denoted by E'—are still qualitatively convincing, see **row 3**, suggesting that the models learn to add a positive bias to all classes, which vanishes in the mean-corrected explanations. To alleviate this, we propose to optimise the models differently (yielding M', see **row 4**) instead of changing the explanations: specifically, by changing the target encoding, as described in Section 6.1.2.2, we encourage the models to focus on positive evidence.

6.4 CONCLUSION

We presented a novel approach for endowing deep neural networks with a high degree of *inherent interpretability*. In particular, we developed the B-cos transformation as a modification of the linear transformation to increase weight-input alignment during optimisation and showed that this can significantly increase interpretability. Importantly, the B-cos transformations can be used as a drop-in replacement for the ubiquitously used linear transformations in conventional DNNs whilst only incurring minor drops in classification accuracy. As such, our approach can increase the interpretability of a wide range of DNNs at a low cost and thus holds great potential to have a significant impact on the deep learning community. In particular, it shows that strong performance and interpretability need not be at odds. Moreover, we demonstrate that by structurally constraining *how* the neural networks are to solve an optimisation task—in the case of B-cos Networks via *alignment*—allows for extracting explanations that faithfully reflect the underlying model. We believe this to be an important step on the road towards interpretable deep learning, which is essential for building trust in DNN-based decisions, especially in safety-critical situations.

III

FROM UNDERSTANDING MODELS TO GUIDING MODELS

Thus far, we have focused on how both post-hoc (Part I) and model-inherent (Part II) explanations can help us better understand the decision process of Deep Neural Networks (DDNs), i.e., our goal was to *extract knowledge* from DNNs. In the following chapter, we aim to do the opposite: specifically, we explore how such explanations can be used to *insert knowledge* into the DNNs, thus ensuring that they rely on the right features for their predictions.

For this, **in Chapter 7**, we present a detailed study on how to guide models by regularising the importance attribution maps obtained for them via a given explanation method. In particular, we analyse the impact of various important design choices on the *effectiveness* of model guidance, such as the formulation of the loss function, the choice of attribution method, or the architecture to which model guidance is applied.

Further, as annotation costs for model guidance might otherwise limit its applicability, we also place a particular focus on *efficiency*. Specifically, we evaluate the robustness of guidance to limited or overly coarse annotations. Here, we find that even annotating only 1% of the training images can lead to significant improvements in object localisation and model generalisation.

Additionally, **in Chapter 8**, we explore whether explanations can improve the transfer of knowledge between DNNs during knowledge distillation (KD).

In KD, a 'student' model is typically trained to produce similar output distributions as a 'teacher' model. We show that additionally optimising for explanation similarity consistently provides large gains in terms of accuracy and student-teacher agreement, ensures that the student learns from the teacher to be right for the right reasons and to give similar explanations, and is robust with respect to the model architectures, the amount of training data, and even works with 'approximate', pre-computed explanations.

7 Using Explanations to Guide Models

Contents

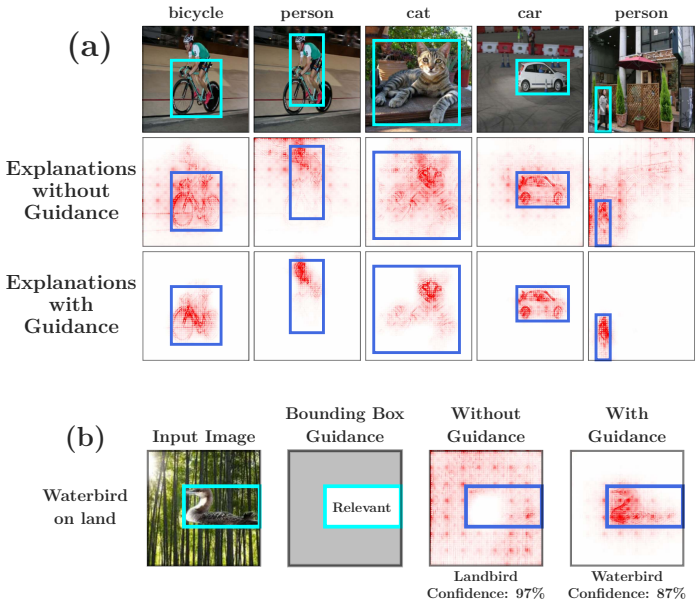


Figure 7.1: **(a) Model guidance increases object focus.** Models may rely on irrelevant background features or spurious correlations (e.g. presence of person provides positive evidence for bicycle, centre row, col. 1). Guiding the model via bounding box annotations can mitigate this and consistently increases the focus on object features (bottom row). **(b) Model guidance can improve accuracy.** In the presence of spurious correlations in the training data, non-guided models might focus on the wrong features. In the example image in (b), the waterbird is incorrectly classified to be a landbird due to the background (col. 3). Guiding the model via bounding box annotation (as shown in col. 2), the model can be guided to focus on the bird features for classification (col. 4).

While Deep Neural Networks (DNNs) excel at learning features that are highly predictive on a set of training images, these features do not always generalise to unseen images, as DNNs might also memorise individual images [FZ20] or rely on spurious correlations in the data [XEIM21]. E.g., if bikes are highly correlated with people during training, a model might learn to use the presence of a person as evidence for a bike (e.g. Figure 7.1a, col. 1, rows 1-2), which can limit how well it generalises. Similarly, a bird classifier might rely on features from the bird's habitat, and thus fail to correctly classify instances of the same bird in a different habitat (cf. Figure 7.1b cols. 1-3, [PDN+22]).

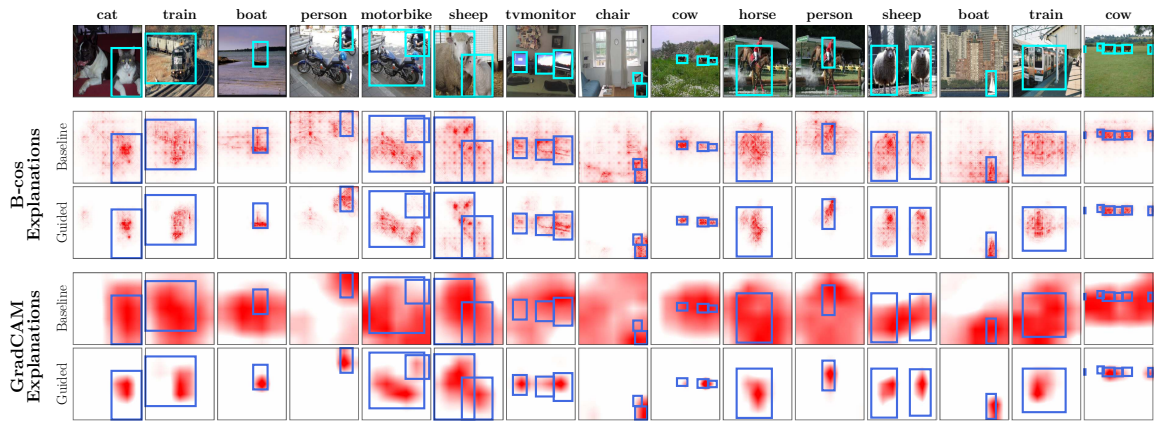


Figure 7.2: **Qualitative results of model guidance.** We show model-inherent B-cos explanations (input layer) of a B-cos ResNet-50 and GradCAM explanations (final layer) of a conventional ResNet-50 before ('Standard') and after optimisation ('Guided') for images from the VOC test set, using our proposed Energy loss (Equation (7.6)). Guiding the model via bounding box annotations consistently increases the focus on object features for both methods. Specifically, we find that background attributions are consistently suppressed in both cases.

To avoid such behaviour, *model guidance* approaches have recently gained popularity [RHDV17, SLL+21, GSZH22, GSB+22, TK19, Tes19]. These take advantage of the fact that many attribution methods are themselves differentiable (e.g. [SGK17, STY17, SCD+17, BFS22b]), and can thus be optimised to guide models to be "right for the right reasons" [RHDV17], e.g. by jointly optimising for both correct classification and for attributing importance to regions deemed relevant by humans. This can help the model focus on the relevant features of a class, and correct errors in reasoning (Figure 7.1b, col. 4). Such guidance has the added benefit of providing well-localised explanations that are thus easier to understand for end users (e.g. Figure 7.2).

While model guidance has shown promising results, a detailed study of how to do this most *effectively* is crucially missing. In particular, model guidance has so far been studied for a limited set of attribution methods and models and usually on relatively simple and/or synthetic datasets; further, the evaluation settings between approaches can significantly differ, which makes a fair comparison difficult.

Therefore, in this chapter, we perform an in-depth evaluation of model guidance on large scale, real-world datasets, to better understand the effectiveness of a variety of design choices. Specifically, we evaluate model guidance along the following dimensions: the model architecture, the guidance *depth*[1], the attribution method, and the loss function. In this context, we propose using the EPG score [WWD+20]—an evaluation metric that has thus far been used to evaluate the quality of attribution methods—as an additional loss function (which we call the Energy loss) as it is fully differentiable.

Further, as annotation costs can be a major hurdle for making model guidance practical, we place a particular focus on *efficient* guidance. Specifically, we use bounding boxes instead of semantic segmentation masks, and evaluate the robustness of guidance techniques under limited or overly coarse annotations to reduce data collection costs.

In summary, in this chapter we make the following contributions:

(1) We perform an in-depth evaluation of model guidance on challenging large scale, multi-label classification datasets (PASCAL VOC 2007 [EVGW+09], MS COCO 2014 [LMB+14]), assessing the impact

[1]The layer at which guidance is applied, e.g. typically at the last convolutional layer for GradCAM [SCD+17] or the first layer for IxG [SGK17].

of attribution methods, model architectures, guidance depths, and loss functions. Further, we show that, despite being relatively coarse, bounding box supervision can provide sufficient guidance to the models whilst being much cheaper to obtain than semantic segmentation masks.

(2) We propose using the Energy Pointing Game (EPG) score [WWD+20] as an alternative to the IoU metric for evaluating the effectiveness of such guidance and show that the EPG score constitutes a good loss function for model guidance, particularly when using bounding boxes.

(3) We show that model guidance can be performed cost-effectively by using annotation masks that are noisy or are available for only a small fraction (e.g. 1%) of the training data.

(4) We show through experiments on the Waterbirds-100 dataset [SKHL20, PDN+22] that model guidance with a small number of annotations suffices to improve the model's generalisation under distribution shifts at test time.

This chapter is based on [RBPAS23] and the corresponding code is publicly available at: github.com/sukrutrao/Model-Guidance.

7.1 MODEL GUIDANCE

In this section, we introduce the model guidance approach that jointly optimises for classification and localisation (Section 7.1.1). We then describe the attribution methods (Section 7.1.2), metrics (Section 7.1.3), and localisation loss functions (Section 7.1.4) that we evaluate in Section 7.3. Finally, in Section 7.1.5 we discuss how we train for localisation in the presence of multiple ground truth classes.

Notation. We consider a multi-label classification problem with K classes with $X \in \mathbb{R}^{C\times H\times W}$ the input image and $y \in \{0,1\}^K$ the one-hot encoding of the image labels. With $A_k \in \mathbb{R}^{H\times W}$ we denote an attribution map for a class k for X using a classifier f; A_k^+ denotes the positive component of the attributions, $\hat{A}_k = \frac{A_k}{\max(\text{abs}(A_k))}$ normalised attributions, and $\hat{A}_k^+ = \frac{A_k^+}{\max(A_k^+)}$ normalised positive attributions. Finally, $M_k \in \{0,1\}^{H\times W}$ denotes the binary mask for class k, which is given by the union of bounding boxes of all occurrences of class k in X.

7.1.1 General Definition

Following prior work (e.g. [RHDV17, SLL+21, GSZH22, GSB+22]), the model is trained jointly for classification and localisation (cf. Figure 7.3):

$$\mathcal{L} = \mathcal{L}_{\text{class}} + \lambda_{\text{loc}}\mathcal{L}_{\text{loc}} \ . \tag{7.1}$$

I.e., the loss consists of a classification loss ($\mathcal{L}_{\text{class}}$), for which we use binary cross-entropy, and a localisation loss ($\mathcal{L}_{\text{loc}}$), which we discuss in Section 7.1.4; here, the hyperparameter λ_{loc} controls the weight given to each of the objectives.

7.1.2 Attribution Methods

In contrast to prior work that typically use GradCAM [SCD+17] attributions, we perform an evaluation over a selection of popularly used differentiable[2] attribution methods which have been shown to localise well [RBS22]: IxG [SGK17], IntGrad [STY17], and GradCAM [SCD+17]. We further evaluate the model-inherent explanations of the B-cos models introduced in the last chapter, see also [BFS22b]. To ensure comparability across attribution methods [RBS22], we evaluate all attribution methods at the input, various intermediate, and the final spatial layer.

IxG [SGK17] computes the element-wise product $\odot$ of the input and the gradients of the k-th output w.r.t. the input, i.e. $X \odot \nabla_X f_k(X)$. For piece-wise linear models such as DNNs with ReLU activations [NH10], this faithfully computes the linear contributions of a given input pixel to the model output.

GradCAM [SCD+17] computes importance attributions as a ReLU-thresholded, gradient-weighted sum of activation maps. In detail, it is given by $\text{ReLU}(\sum_c \alpha_c^k \odot U_c)$ with c denoting the channel dimension, and α^k the average-pooled gradients of the output for class k with respect to the activations U of the last convolutional layer in the model.

IntGrad [STY17] takes an axiomatic approach and is formulated as the integral of gradients over a straight line path from a baseline input to the given input X. Approximating this integral requires several

[2] Differentiability is necessary for optimising attributions via gradient descent, so non-differentiable methods (e.g. [RSG16, PDS18]) are not considered.

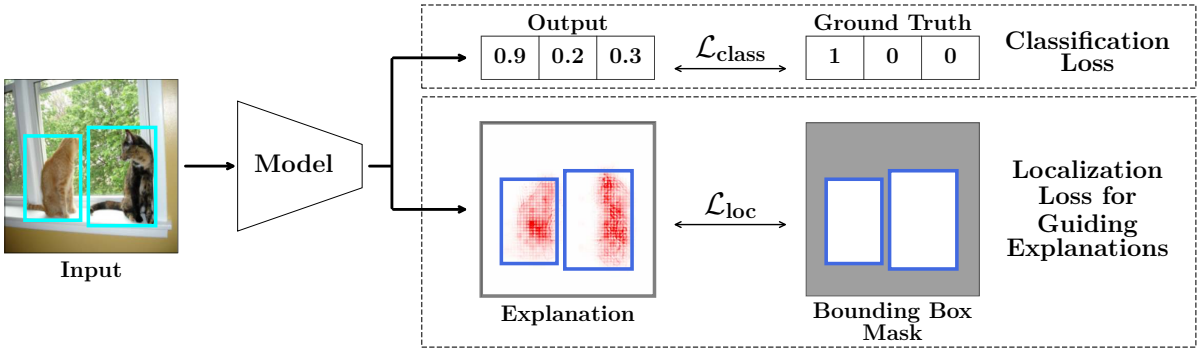


Figure 7.3: **Model guidance overview.** We jointly optimise for classification ($\mathcal{L}_{\text{class}}$) and localisation of attributions to human-annotated bounding boxes ($\mathcal{L}_{\text{loc}}$), to guide the model to focus on object features. Various localisation loss functions can be used, see Section 7.1.4.

gradient computations, making it computationally expensive for use in model guidance. To alleviate this, when optimising with IntGrad, we use the recently proposed $\mathcal{X}$-DNN models [HSMR21] that allow for an exact computation of IntGrad in a single backward pass.

B-cos [BFS22b] attributions are generated using the inherently interpretable B-cos Networks introduced in Chapter 6, which promote alignment between the input $\mathbf{x}$ and a dynamic weight matrix $\mathbf{W}(\mathbf{x})$ during optimisation. We use the contribution maps given by the element-wise product of the dynamic weights with the input ($\mathbf{W}_k^T(\mathbf{x}) \odot \mathbf{x}$), which faithfully represent the contribution of each pixel to class k. To be able to guide B-cos models, for the work in this chapter we additionally developed a differentiable implementation of B-cos explanations, see Appendix D.

7.1.3 Evaluation Metrics

We evaluate the models' performance on two objectives: classification and localisation. For classification, we use the F1 score and mean average precision (mAP). We discuss the localisation metrics below.

Intersection over Union (IoU) is a commonly used metric (cf. [GSB+22]) that computes the intersection between the ground truth annotation masks and the binarised attribution maps, normalised by their union; for binarisation, a threshold parameter needs to be chosen. In this work, the ground truth masks are taken to be the union of all bounding boxes of a class in the image and, following prior work [FV17], the threshold parameter is selected via a heldout set.

Energy-based Pointing Game (EPG) [WWD+20] measures the concentration of attribution energy within the mask, i.e. the fraction of positive attributions inside the bounding boxes:

$$\text{EPG}_k = \frac{\sum_{h=1}^{H}\sum_{w=1}^{W} M_{k,hw} A^+_{k,hw}}{\sum_{h=1}^{H}\sum_{w=1}^{W} A^+_{k,hw}} \,. \tag{7.2}$$

In contrast to IoU, EPG more faithfully takes into account the relative importance given to each input region, since it does not binarise the attributions. Like IoU, the scores lie in $[0, 1]$, with higher scores indicating better localisation.

7.1.4 Localisation Losses

We evaluate the most commonly used localisation losses ($\mathcal{L}_{\text{loc}}$ in Equation (7.1)) from prior work. We describe these losses as applied on attribution maps of an image for a single class k, as well as the proposed EPG-derived Energy loss.

L_1 Loss ([GSZH22, GSB+22], Equation (7.3)) minimises the L_1 distance between annotation masks and normalised positive attributions $\hat{A}^+_k$, guiding the model towards uniform attributions inside the mask and suppressing attributions outside of it.

$$\mathcal{L}_{\text{loc},k} = \frac{1}{H \times W} \sum_{h=1}^{H} \sum_{w=1}^{W} \| M_{k,hw} - \hat{A}^+_{k,hw} \|_1 \tag{7.3}$$

Per-pixel Cross Entropy (PPCE) Loss ([SLL+21], Equation (7.4)) applies a binary cross entropy loss between the mask and the normalised positive annotations $\hat{A}^+_k$, thus guiding the model to maximise the attributions inside the mask:

$$\mathcal{L}_{\text{loc},k} = -\frac{1}{\|M_k\|_1} \sum_{h=1}^{H} \sum_{w=1}^{W} M_{k,hw} \log(\hat{A}^+_{k,hw}) \ . \tag{7.4}$$

As PPCE does not constrain attributions outside the mask, there is no explicit pressure to avoid spurious features.

RRR* Loss ([RHDV17], Equation (7.5)). [RHDV17] introduced the RRR loss to regularise the normalised input gradients $\hat{A}_{k,hw}$ as

$$\mathcal{L}_{\text{loc},k} = \sum_{h=1}^{H} \sum_{w=1}^{W} (1 - M_{k,hw}) \hat{A}^2_{k,hw} \ . \tag{7.5}$$

To extend it to our setting, we take $\hat{A}_{k,hw}$ to be given by an arbitrary attribution method (e.g. IntGrad); we denote this generalised version by RRR*. In contrast to the PPCE loss, RRR* only regularises attributions *outside* the ground truth masks. While it thus does not introduce a uniformity prior similar to the L_1 loss, it also does not explicitly promote high importance attributions inside the masks.

Energy Loss. In addition to the losses described in prior work, we propose to also evaluate using the EPG score ([WWD+20], Equation (7.2)) as a loss function for model guidance, as it is fully differentiable. In particular, we simply define it as

$$\mathcal{L}_{\text{loc},k} = -\text{EPG}_k. \tag{7.6}$$

Unlike existing localisation losses that either (i) do not constrain attributions across the entire input (RRR*, PPCE), or (ii) force the model to attribute uniformly within the mask even if it includes irrelevant background regions (L_1, PPCE), maximising the EPG score jointly optimises for higher attribution energy within the mask and lower attribution energy outside the mask. By not enforcing a uniformity prior, we find that the Energy loss is able to provide effective guidance while allowing the model to learn freely what to focus on within the bounding boxes (Section 7.3).

7.1.5 Efficient Optimisation

In contrast to prior work [RHDV17, SLL+21, GSZH22, GSB+22], we perform model guidance on a multi-label classification setting, and consequently there are multiple ground truth classes whose attribution localisation could be optimised. Computing and optimising for several attributions within an image would add a significant overhead to the computational cost of training (multiple backward passes). Hence, for

efficiency, we sample one ground truth class k per image at random for every batch and only optimise for localisation of that class, i.e., $\mathcal{L}_{\text{loc}} = \mathcal{L}_{\text{loc},k}$. We find that this still provides effective model guidance while keeping the training cost tractable.

7.2 EXPERIMENTAL SETUP

In this section, we describe our experimental setup and how we select the best models across metrics; for full details, see Appendix D. We evaluate across all possible choices for each category, and discuss our results in Section 7.3.

Datasets. We evaluate on PASCAL VOC 2007 [EVGW+09] and MS COCO 2014 [LMB+14] for multi-label image classification. In Section 7.3.5, to understand the effectiveness of model guidance in mitigating spurious correlations, we also evaluate on the synthetically constructed Waterbirds-100 dataset [SKHL20, PDN+22], where landbirds are perfectly correlated with land backgrounds on the training and validation sets, but are equally likely to occur on land or water in the test set (similar for waterbirds and water). With this dataset, we evaluate model guidance for suppressing undesired features.

Attribution Methods and Architectures. As described in Section 7.1.2, we evaluate with IxG [SGK17], IntGrad [STY17], B-cos [BFS22b, BSFS24], and GradCAM [SCD+17] using models with a ResNet-50 [HZRS16] backbone. For IntGrad, we use an $\mathcal{X}$-DNN ResNet-50 [HSMR21] to reduce the computational cost, and a B-cos ResNet-50 for the B-cos attributions. To emphasise that the results generalise across different backbones, we further provide results for a B-cos ViT-S [DBK+21, BSFS24] and a B-cos DenseNet-121 [HLVDMW17, BSFS24]. We evaluate optimising the attributions at different network layers, such as at the input image and the last convolutional layers' output[3], as well as at multiple intermediate layers. Within Chapter 7, we highlight some of the most representative and insightful results, the full set of results can be found in Appendix D. All models were pretrained on ImageNet [DDS+09], and model guidance was applied when fine-tuning the models on the target dataset.

Localisation Losses. As described in Section 7.1.4, we compare four localisation losses in our evaluation: Energy, L_1 [GSZH22, GSB+22], PPCE [SLL+21], and RRR* (cf. Section 7.1.4, [RHDV17]).

Evaluation Metrics. As discussed in Section 7.1.3, we evaluate both for classification and localisation performance of the models. For classification, we report the F1 scores, similar results with mAP scores can be found in Appendix D. For localisation, we evaluate using the EPG and IoU scores.

Selecting the Best Models. As we evaluate for two distinct objectives (classification + localisation), it is not trivial to decide which models perform 'the best', e.g. a model that provides the best classification performance might provide significantly worse localisation than a model that provides only slightly lower classification performance. Finding the right balance and deciding which of those models in fact constitutes the 'better' model depends on the preference of the end user. Hence, instead of selecting models based on a single metric, we select the set of Pareto-dominant models [Par94, Par08, Bac80] across three metrics—F1, EPG, and IoU—for each training configuration, as defined by a combination of attribution method, layer, and loss. Specifically, as shown in Figure 7.4, we train each configuration using three different choices of λ_{loc}, and select the set of Pareto-dominant models among all checkpoints (epochs and λ_{loc}). This provides a more holistic view of the general trends on the effectiveness of model guidance for each configuration.

[3] As typically used in IxG (input) and GradCAM (final) respectively.

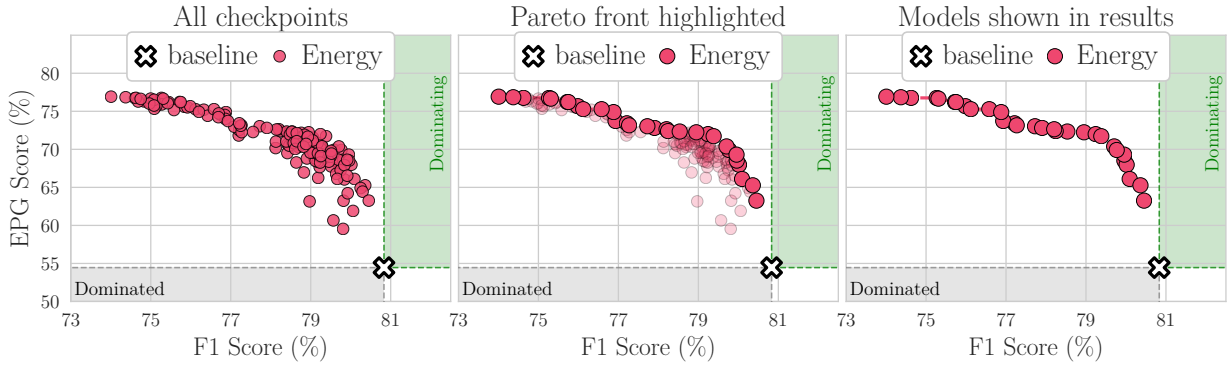


Figure 7.4: **Selecting models for evaluation.** For each configuration, we evaluate every model at every checkpoint and measure its performance across various metrics (F1, EPG, IoU) on the validation set; i.e. every point in the left graph corresponds to one model (for B-cos models optimised via the Energy loss at the input layer). Instead of evaluating a single model on the test set, we evaluate *all Pareto-dominant* models, see centre and right plot.

7.3 Experimental Results

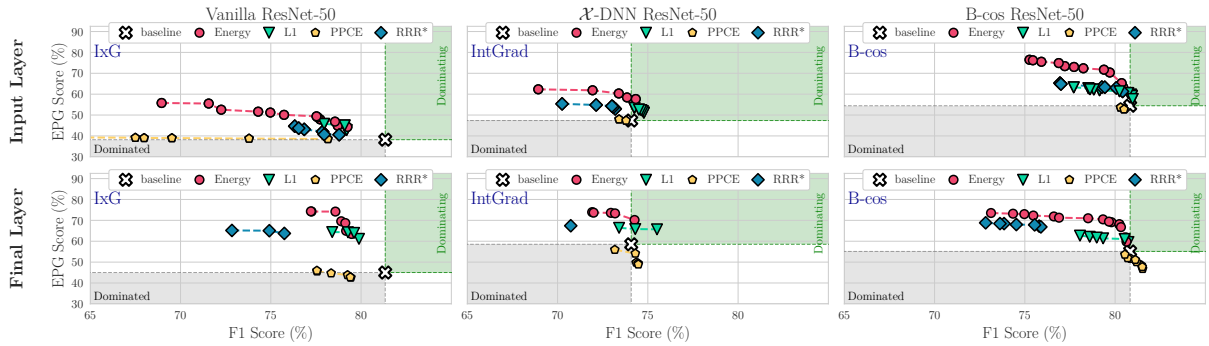
In this section, we discuss our experimental findings. In particular, in Section 7.3.1, we discuss the impact of the loss functions on the EPG and IoU scores of the models; in Section 7.3.2, we analyse the impact of the models and attribution methods; in Section 7.3.3, we show that guiding the models via their explanations can lead to improved classification accuracy. In Section 7.3.4, we present additional studies in which we evaluate and discuss the cost of model guidance approaches: in particular, we study model guidance with limited additional labels, with increasingly coarse bounding boxes, and at deep layers in the network. Finally, in Section 7.3.5, we show the utility of model guidance in improving accuracy in the presence of distribution shifts. For easier reference, we label our individual findings as **R1**–**R9**.

Note. To draw conclusive insights and highlight general and reliable trends in the experiments, we compare the Pareto curves (see Figure 7.4) of individual configurations. If the Pareto curve of a specific loss (e.g. Energy in Figure 7.5) consistently Pareto-dominates the Pareto curves of all other losses, we can conclude that for the combination of evaluated metrics (e.g. EPG vs. F1), this loss is the best choice.

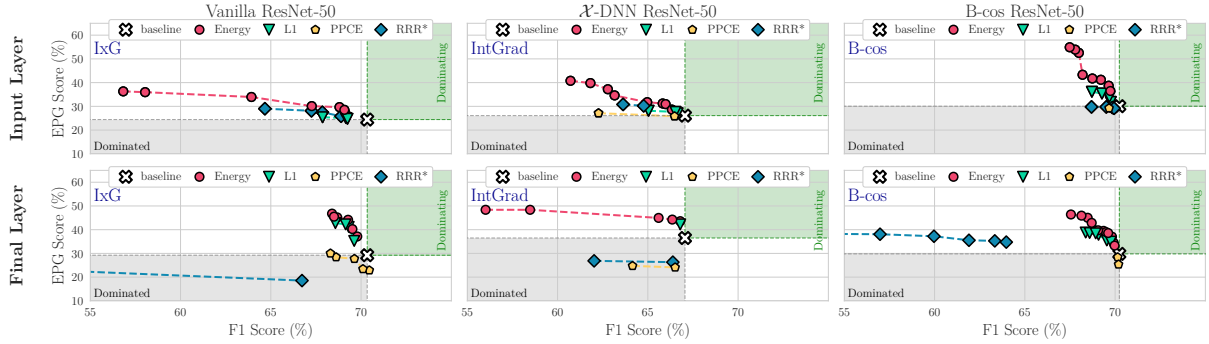
7.3.1 Comparing Loss Functions for Model Guidance

In the following, we highlight the main insights gained from the *quantitative* evaluations. For a *qualitative* comparison between the losses, please see Figure 7.9; note that we show examples for a B-cos model as the differences become clearest; full results can be found in Appendix D.

R1 The Energy Loss Yields the Best EPG Scores. In Figure 7.5, we plot the Pareto curves for EPG vs. F1 scores for a wide range of configurations (see Section 7.2) on VOC (a) and COCO (b); specifically, we group the results by model type (Vanilla, $\mathcal{X}$-DNN, B-cos), the layer depths at which the attribution was regularised (Input / Final), and the loss used during optimisation (Energy, L_1, PPCE, RRR*). From these results it becomes apparent that the optimisation with the Energy loss yields the best trade-off between accuracy (F1) and the EPG score: e.g., when looking at the upper right plot in Figure 7.5a we can see that the Energy loss (red dots) improves over the baseline B-cos model (white cross) by improving the localisation in terms of EPG score with only a minor cost in classification performance



(a) **PASCAL VOC results for EPG vs. F1.**



(b) **MS COCO results for EPG vs. F1.**

Figure 7.5: **EPG vs. F1,** for different datasets (**(a)**: VOC; **(b)**: COCO), losses (**markers**) and models (**columns**), optimised at different layers (**rows**); additionally, we show the performance of the baseline model before fine-tuning and demarcate regions that strictly dominate (are strictly dominated by) the baseline performance in green (grey). For each configuration, we show the Pareto fronts (cf. Figure 7.4) across regularisation strengths λ_{loc} and epochs (cf. Section 7.3 and Figure 7.4). We find the Energy loss to give the best trade-off between EPG and F1.

(i.e. F1 score). Further trading off F1 scores yields even higher EPG scores. Importantly, the Energy loss Pareto-dominates all the other losses (RRR*: blue diamonds; L_1: green triangles; PPCE: yellow pentagons). This is is also true for the other network types (Vanilla ResNet-50, Figure 7.5a (top left), and $\mathcal{X}$-DNN, Figure 7.5a (top centre)) and at the final layer (bottom row), and generalises across backbone architectures (Figure 7.7). When comparing Figure 7.5a and Figure 7.5b, we also find these results to be highly consistent between datasets.

R2 The L_1 Loss Yields the Best IoU Performance. Similarly, in Figure 7.6, we plot the Pareto curves of IoU vs. F1 scores for various configurations at the final layer; for the IoU results at the input layer and on the COCO dataset, please see Appendix D. For IoU, the L_1 loss provides the best trade-off and, with few exceptions, L_1-guided models Pareto-dominate all other models in all configurations.

R3 The Energy Loss Focuses Best on On-object Features. By not forcing the models to highlight the entire bounding boxes (see Section 7.1.4), we find that the Energy loss also suppresses background features *within* the bounding boxes, thus better preserving fine details of the explanations (cf. Figures 7.9 and 7.11). To quantify this, we evaluate the distribution of Energy (Equation (7.2)) just within the bounding boxes. For this, we take advantage of the segmentation mask annotations available for a subset of the VOC test set. Specifically, we measure the Energy contained in the segmentation masks versus the entire bounding

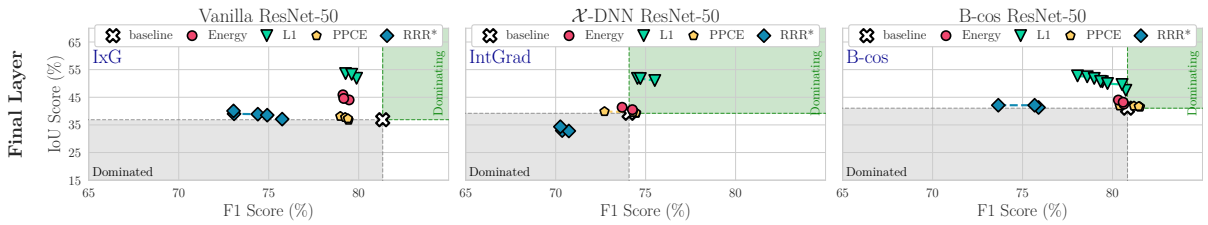


Figure 7.6: **IoU vs. F1,** for different losses (**markers**) and models (**columns**) for VOC; results for COCO are in Appendix D. Additionally, we show the performance of the baseline model before fine-tuning and demarcate regions that strictly dominate (are strictly dominated by) the baseline model in green (grey). For each configuration, we show the Pareto fronts (Figure 7.4) across regularisation strengths λ_{loc} and all epochs; for details, see Sections 7.2 and 7.3. Across all configurations, we find the L_1 loss to provide the largest gains in IoU at the lowest cost.

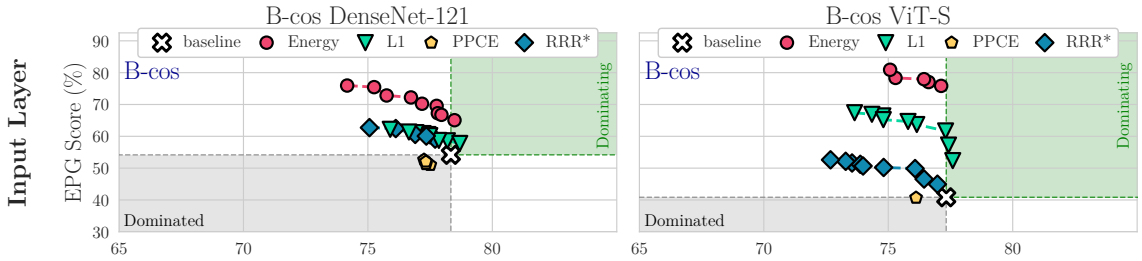


Figure 7.7: **EPG vs. F1 on VOC.** We observe the same trends as in Figure 7.5a for different backbone architectures, specifically a B-cos DenseNet-121 and a B-cos ViT-S. For IoU results, see Appendix D.

box, which indicates how much of the attributions actually highlight on-object features. We find that the Energy loss outperforms L_1 across all models and configurations; see Appendix D for details.

In short, we find that the Energy loss works best for improving the EPG metric, whereas the L_1 loss yields the highest gains in terms of IoU; depending on the use case, either of these losses could thus be recommendable. However, we find that the Energy loss is more robust to annotation errors (**R8**, Section 7.3.4), and, as discussed in **R3**, the Energy loss more reliably focuses on object-specific features.

7.3.2 Comparing Models and Attribution Methods

In the following, we highlight our findings regarding different attribution methods and models. Given the similarity of the results between GradCAM and IxG, and since B-cos attributions performed better than GradCAM for B-cos models, we show GradCAM results in Appendix D.

R4 At the Input Layer, B-cos Explanations Perform Best. The B-cos models not only achieve the highest EPG/ IoU performance before applying model guidance, ('baselines') but also obtain the highest gains in EPG and IoU and thus the highest overall performance (for EPG see Figure 7.5, right; for IoU, see Appendix D): e.g., an Energy-based B-cos model achieves an EPG score of 71.7 @ 79.4% F1, thus significantly outperforming the best EPG scores of both other model types at a lower cost in F1 (Vanilla: 55.8 @ 69.0%, $\mathcal{X}$-DNN: 62.3 @ 68.9%). This is also observed *qualitatively*, as we show in Appendix D.

R5 Regularising at the Final Layer Yields Consistent Gains. As can be seen in Figure 7.5 (bottom) and Figure 7.6, all models can be guided well via regularisation at the final layer, i.e. all models show improvements in IoU and EPG score.

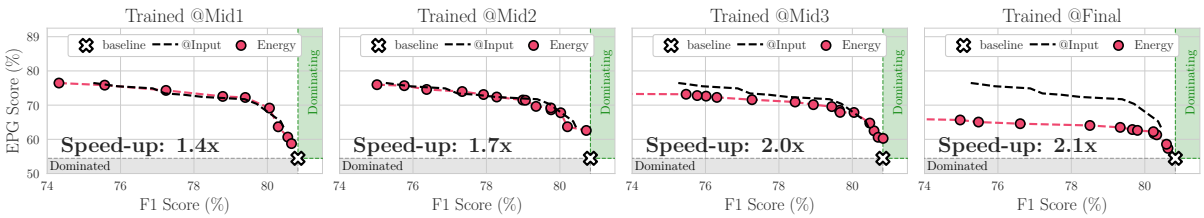


Figure 7.8: **Faster training by guiding at later layers.** While input-level attributions tend to be more detailed (cf. Figure 7.2), they are costlier to compute than attributions at later layers. However, we find that guidance at later layers (e.g. @Mid3) also significantly improves input-level attributions, yielding similar EPG results as input-level guidance (@Input) at up to twice the training speed; for IoU results, see Appendix D.

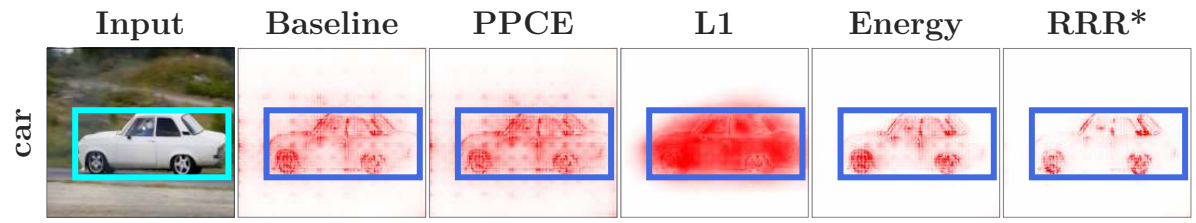


Figure 7.9: **Loss comparison** for input attributions of a B-cos model. We show attributions before (baseline, col. 2) and after guidance (cols. 3-6) for a specific image (col. 1) and its bounding box annotation. We find that the Energy and the RRR* losses yield sparse attributions, whereas the L_1 loss yields smoother attributions, as it is optimised to fill the entire bounding box. For the PPCE loss we observe only a minor effect on the attributions.

In short, we find model guidance to work well across all tested models when optimising at the final layer (**R5**), highlighting its wide applicability. However, the inherent explanations of B-cos models yield more detailed and well-localised attributions at the input layer (**R4**).

7.3.3 Improving Accuracy with Model Guidance

R6 Model Guidance can Improve Accuracy. For both the Vanilla models (final layer) and the $\mathcal{X}$-DNNs (input+final), we found models that improve the localisation metrics *and* the F1 score. These improvements are particularly pronounced for the $\mathcal{X}$-DNN: e.g., we find models that improve the EPG and F1 scores by $\Delta=7.2$ p.p. and $\Delta=1.4$ p.p. respectively (Figure 7.5, centre top), or the IoU and F1 scores by $\Delta=11.9$ p.p. and $\Delta=1.4$ p.p. (Figure 7.6, centre).

However, overall we observe a trade-off between localisation and accuracy (Figures 7.5 and 7.6). Given the similarity of the training and test distributions, focusing on the object need not improve classification performance, as spurious features are also present at test time, and discouraging the guided model relying on contextual features makes the classification more challenging. In Section 7.3.5, we show that guidance can significantly improve performance when there is a distribution shift between training and test.

7.3.4 Efficiency and Robustness Considerations

While bounding boxes decrease the data collection cost with respect to segmentation masks, they can nonetheless be expensive to obtain, especially when expert knowledge is required. To further reduce

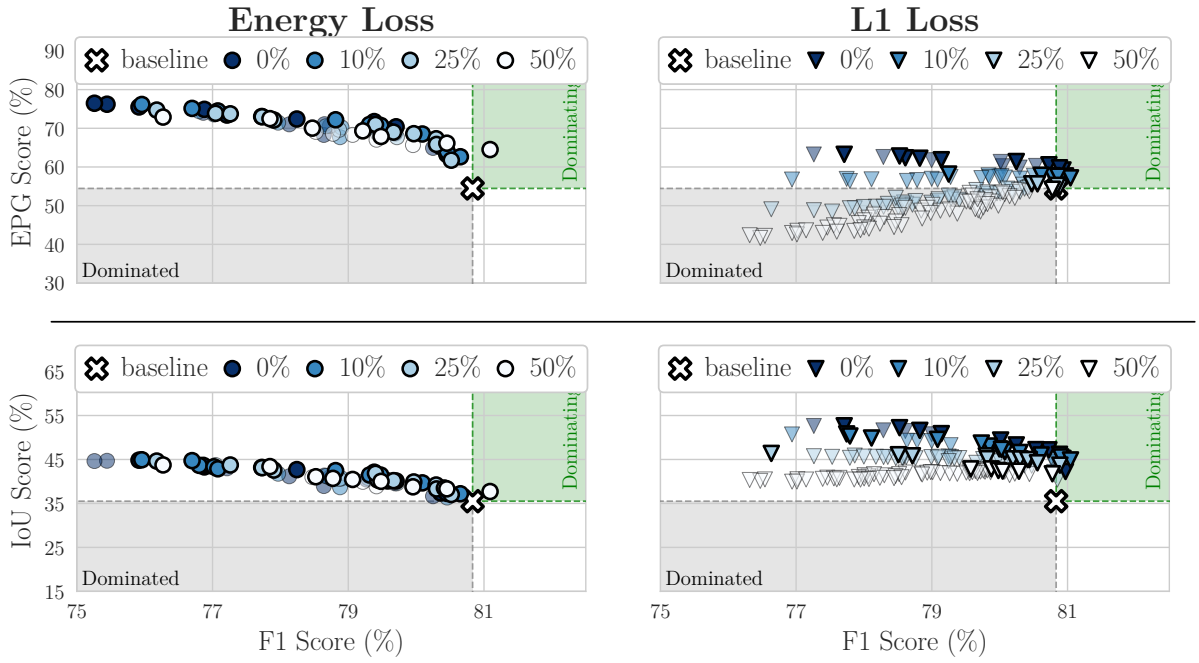


Figure 7.10: **Quantitative results for dilated bounding boxes** for a B-cos model at the input layer. We show EPG and IoU (top and bottom) results for models trained with various amounts of annotation errors (increasingly large bounding boxes, see Figure 7.11). The Energy loss yields highly consistent results despite training with heavily dilated bounding boxes (left), whereas the results of the L_1 loss (right) worsen markedly; best viewed in colour.

those costs, in this section, we assess the robustness of guiding the model with a limited number (**R7**) or increasingly coarse annotations (**R8**). Apart from *data efficiency*, we further explore how *training efficiency* can be improved for fine-grained (i.e. input-level) explanations (**R9**), as explanations at early layers are more costly to obtain than those at later layers.

R7 Model Guidance Requires Only Few Additional Annotations. In Figure 7.12, we show that the EPG score can be significantly improved with a very limited number of annotations; for IoU results, see Appendix D. Specifically, we find that when using only 1% of the training data (25 annotated images) for VOC, improvements of up to $\Delta{=}23.0$ p.p. ($\Delta{=}1.4$) in EPG (IoU) can be obtained, at a minor drop in F1 ($\Delta{=}0.3$ p.p. and $\Delta{=}2.5$ p.p. respectively). When annotating up to 10% of the images, very similar results can be achieved as with full annotation (see e.g. cols. 2+3 in Figure 7.12).

R8 The Energy Loss is Highly Robust to Annotation Errors. As discussed in Section 7.1.4, the Energy loss only directs the model on which features *not* to use and does not impose a uniform prior on the attributions within the bounding boxes. As a result, we find it to be much more stable to annotation errors: e.g., in Figure 7.10, we visualise how the EPG (top) and IoU (bottom) scores of the best performing models under the Energy (left) and L_1 loss (right) evolve when using coarser bounding boxes; for this, we simply dilate the bounding box size by $p{\in}\{10, 25, 50\}\%$ during training, see Figure 7.11. While the models optimised via the L_1 loss achieve increasingly worse results (right), the Energy-optimised models are essentially unaffected by the coarseness of the annotations.

In short, we find that the models can be guided effectively at a low cost in terms of annotation effort, as only few annotations (e.g. 25 for VOC) are required (cf.**R7**), and, especially for the Energy loss, these annotations can be very coarse and do not have to be 'pixel-perfect' (cf.**R8**).

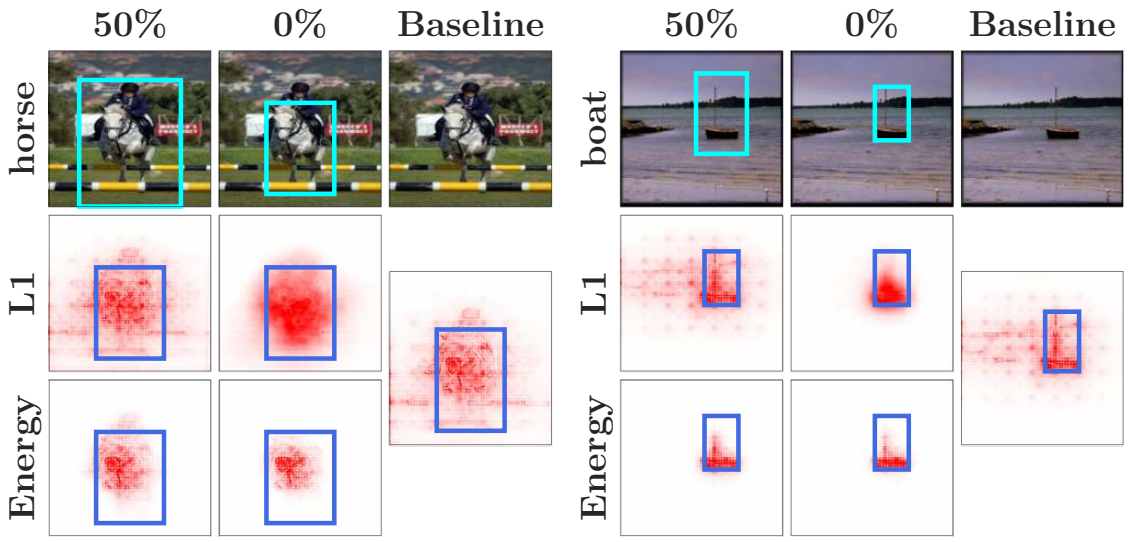


Figure 7.11: **Qualitative results for dilated bounding boxes** for a B-cos model at input. Examples for attributions (rows 2+3) of models trained with dilated bounding boxes (row 1). In contrast to L_1, models trained with Energy show significant gains in object focus even with significant noise (e.g. 'Baseline' vs. '50%').

R9 Guidance at Deep Layers can be Effective. While guided input-level explanations of B-cos Networks exhibit a high degree of detail, regularising those explanations comes at an added training cost. In particular, optimising at the input layer requires backpropagating through the entire network to compute the attributions. In an effort to reduce training costs whilst maintaining the benefits of fine-grained explanations at input resolution, we evaluate if input-level attributions benefit from an optimisation at deeper layers. Specifically, we regularise B-cos attributions at the final and at three intermediate layers (Mid{1,2,3}), and evaluate the localisation of attributions at the input. We find (Figure 7.8) that training at a deeper layer can provide significant speed-ups, often at a negligible cost in localisation performance. E.g., since we do not have to compute a full backward pass through the model during training, optimising at Mid2 (col. 2 in Figure 7.8) provides similar gains in localisation at a 1.7x speed-up in training time.

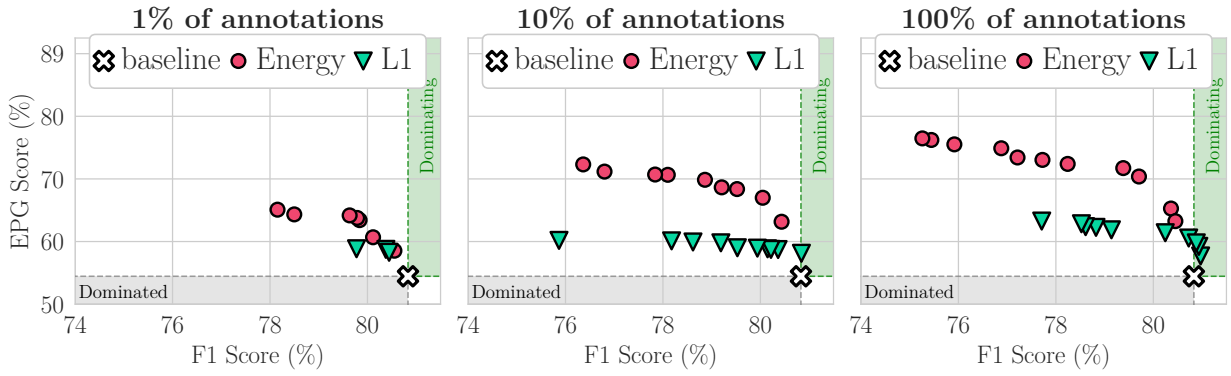


Figure 7.12: **EPG results with limited annotations** for a B-cos model at the input layer, optimised with the Energy and the L_1 loss. Using bounding box annotations for as little as 1% (left) of the images yields significant improvements in EPG, and with 10% (centre) similar gains as in the fully annotated setting (right) are obtained.

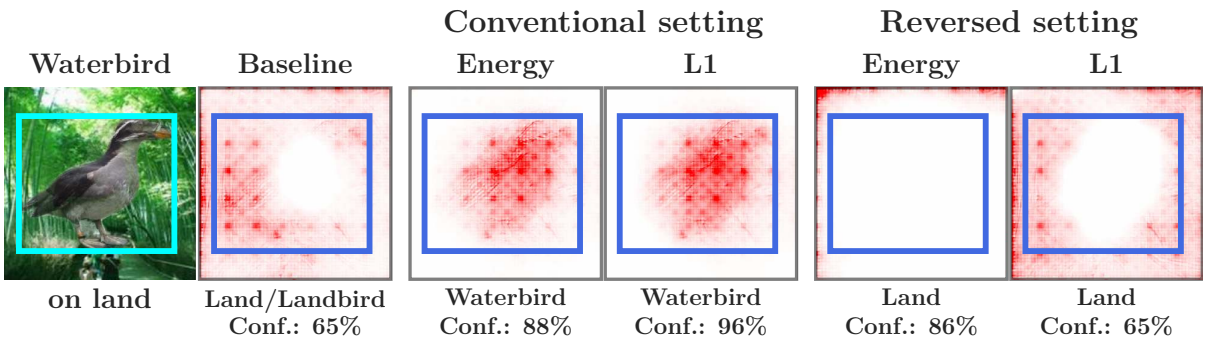


Figure 7.13: **Qualitative Waterbirds-100 results.** Without guidance, a model might focus on the background to classify birds (baseline) and thus misclassify waterbirds on land (col. 2). Guiding models can correct such errors and focus on the desired feature: in cols. 3+4 (5+6) the model is guided to classify by using the bird (background) features and arrives at the desired prediction. Predictions and confidence scores are indicated below the images.

	Conventional		Reversed	
Model	Worst	Overall	Worst	Overall
Baseline	43.4 (±2.4)	68.7 (±0.2)	56.6 (±2.4)	80.1 (±0.2)
Energy	**56.1 (±4.0)**	**71.2 (±0.1)**	**62.8 (±2.1)**	**83.6 (±1.1)**
L_1	51.1 (±1.9)	69.5 (±0.2)	58.8 (±5.0)	82.2 (±0.9)

Table 7.1: **Waterbirds-100 results.** Model guidance is effective in improving both worst-group ('Waterbird on Land') and overall accuracy in the conventional (Landbird vs. Waterbird) and reversed (Land vs. Water) settings. For full results, please see Appendix D.

7.3.5 Effectiveness Against Spurious Correlations

To evaluate the potential for mitigating spurious correlations, we evaluate model guidance with the Energy and L_1 losses on the synthetically constructed Waterbirds-100 dataset [SKHL20, PDN+22]. We perform model guidance under two settings: (1) the conventional setting to classify between landbirds and waterbirds, using the region within the bounding box as the mask; and (2) the reversed setting [PDN+22] to classify the background, i.e., land vs. water, using the region outside the bounding box as the mask. To simulate a limited annotation budget, we only use bounding boxes for a random 1% of the training set, and report results averaged over four runs. We show the results for the worst-group accuracy (i.e., images containing a waterbird on land) and the overall accuracy using B-cos models in Table 7.1; full results for all attributions and models can be found in Appendix D.

Both losses consistently and significantly improve the accuracy in the conventional and the reversed settings by guiding the model to select the 'right' features, i.e. birds (conventional) or background (reversed). This guidance can also be observed qualitatively (cf. Figure 7.13).

7.4 Conclusion

In this chapter, we comprehensively evaluated various models, attribution methods, and loss functions for their utility in guiding models to be "right for the right reasons".

In summary, we find that guiding models via bounding boxes can significantly improve EPG and IoU performance of the optimised attribution method, with the Energy loss working best to improve the EPG

score (R1) and the L_1 loss yielding the highest gains in IoU scores (R2). While the B-cos models achieve the best results in IoU and EPG score at the input layer (R4), all tested model types (Vanilla, $\mathcal{X}$-DNN, B-cos) lend themselves well to being optimised at the final layer (R5), which can even improve attribution maps at early layers (R9). Further, we find that regularising the explanations of the models and thereby 'telling them where to look' can increase the object recognition performance (mAP/accuracy) of some models (R6), especially when strong spurious correlations are present (Section 7.3.5). Interestingly, those gains (EPG, IoU), can be achieved with relatively little additional annotation (R7). Lastly, we find that by not assuming a uniform prior over the attributions within the annotated bounding boxes, training with the energy loss is more robust to annotation errors (R8) and results in models that produce attribution maps that are more focused on class-specific features (R3).

8 GOOD TEACHERS EXPLAIN

Contents

In contrast to providing additional information to models via human annotations, see Chapter 7, in this chapter we investigate whether explanations can also help transfer knowledge *between models*. In particular, we study whether explanations can help improve knowledge distillation (KD) approaches. KD [HVD15] has proven to be an effective scheme for improving classification accuracies of relatively small 'student' models, by training them to match the logit distribution of larger, more powerful 'teacher' models. Despite its simplicity, this approach can be sufficient for the students to match the teacher's accuracy, while requiring only a fraction of the computational resources of the teacher [BZR+22].

Recent findings, however, show that while the students might match the teacher's accuracy, the knowledge is nonetheless not distilled faithfully [SIK+21]. Faithful KD, i.e., a distillation that ensures that the teacher's and the student's functions share properties beyond classification accuracy, is however desirable for many reasons. E.g., the lack of agreement between models [SIK+21] can hurt the user experience when updating machine-learning-based applications [BNK+19, YXK+21]. Similarly, if the students use different input features than the teachers, they might not be *right for the right reasons* [RHDV17]. Further, given the recent AI Act proposal by European legislators [Cou21], it is likely that the interpretability of models will play an increasingly important role and become an intrinsic part of the model functionality. To maintain the *full* functionality of a model, KD should thus ensure that the students allow for the same degree of model interpretability as the teachers.

To address this, in this chapter we study if promoting explanation similarity using commonly used model explanations such as GradCAM [SCD+17] or those of the B-cos models [BFS22b] introduced in Chapter 6 can increase KD fidelity. This should be the case if such explanations indeed reflect meaningful aspects of the models' 'internal reasoning'. Concretely, we propose **'explanation-enhanced' KD (e^{2}KD)**, a simple, parameter-free, and model-agnostic addition to KD in which we train the student to also match the teacher's explanations. Interestingly, despite its simplicity, we find e^{2}KD to significantly improve distillation fidelity in a variety of settings (Figure 8.1). Specifically, e^{2}KD improves student accuracy by ensuring that the students learn to be right for the right reasons and inherently promotes consistent explanations between teachers and students. Moreover, the benefits of e^{2}KD are robust to limited data, approximate explanations, and across model architectures. In summary, we make the following contributions:

(1) We propose **explanation-enhanced KD (e^{2}KD)** and train the students to not only match the teachers'

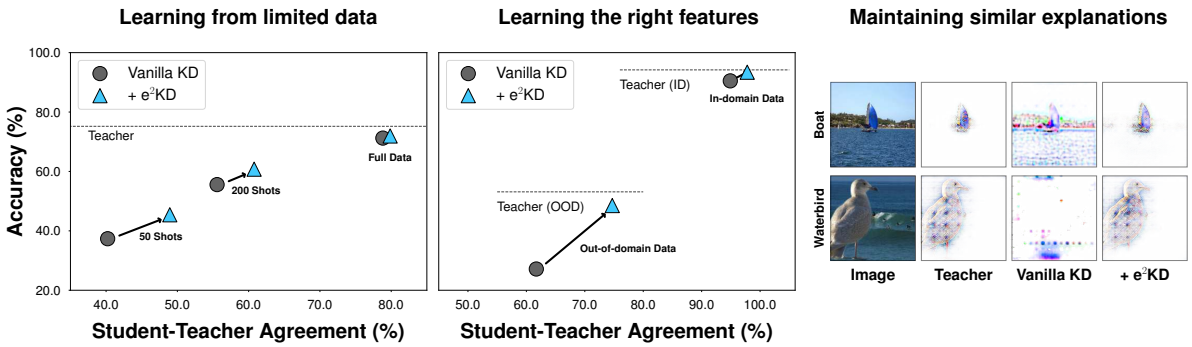


Figure 8.1: **A good teacher explains.** Using explanation-enhanced KD (**e^{2}KD**) can improve KD fidelity and student performance. E.g., e^{2}KD allows the student models to more faithfully approximate the teacher, especially when using fewer data, leading to large gains in accuracy and agreement between teacher and student **(left)**. Further, by guiding the students to give the same explanations as the teachers, e^{2}KD ensures that students learn to be 'right for the right reasons', improving their accuracy under distribution shifts **(centre)**. Lastly, e^{2}KD students learn similar explanations as the teachers, thus exhibiting a similar degree of interpretability as the teacher **(right)** .

logits, but also their explanations (Section 8.1.1). This not only yields highly competitive students in terms of accuracy, but also significantly improves KD fidelity on the ImageNet [DDS$^+$09], Waterbirds-100 [SKHL20, PDN$^+$22], and PASCAL VOC 2007 [EVGW$^+$09] datasets.

(2) We discuss three desiderata for measuring KD fidelity. In particular, we **(a)** evaluate whether the student is performant and has high agreement with the teacher (Section 8.1.2.1), **(b)** examine whether students learn to use the same input features as a teacher that was guided to be 'right for the right reasons' even when distilling with biased data (Section 8.1.2.2), and **(c)** explore whether they learn the same explanations and architectural priors as the teacher (Section 8.1.2.3).

(3) We show e^{2}KD to be a robust approach for improving knowledge distillation, which provides consistent gains across model architectures and with limited data. Moreover, e^{2}KD is even robust to using cheaper 'approximate' explanations. Specifically, for this we propose using '**frozen explanations**' which are only computed once and, during training, we apply any augmentations simultaneously to the image and the explanations (Section 8.1.3).

8.1 EXPLANATION-ENHANCED KD (E^{2}KD) & DISTILLATION FIDELITY

To increase KD fidelity, in Section 8.1.1 we introduce our proposed *explanation-enhanced KD* (e^{2}KD). Further, in Section 8.1.2, we present three desiderata that faithful KD should fulfill and why we expect e^{2}KD to be beneficial in the presented settings. Finally, in Section 8.1.3, we describe how to take advantage of e^{2}KD even without querying the teacher more than once per image when training the student.

Notation. For model M and input x, we denote the predicted class probabilities by $p_M(x)$, obtained using softmax $\sigma(.)$ over output logits $z_M(x)$, possibly scaled by temperature τ. Lastly, we denote the class with highest probability by $\hat{y}_M$.

8.1.1 Explanation-Enhanced Knowledge Distillation (e^{2}KD)

The logit-based knowledge distillation loss $\mathcal{L}_{KD}$ which minimises KL-Divergence D_{KL} between teacher T and student S output probabilities is given by

$$\begin{aligned}\mathcal{L}_{KD} &= \tau^2 D_{\mathrm{KL}}(p_T(x;\tau) \mid\mid p_S(x;\tau)) \\ &= -\tau^2 \sum_{j=1}^{c} \sigma_j\left(\frac{z_T}{\tau}\right) \log \sigma_j\left(\frac{z_S}{\tau}\right).\end{aligned} \tag{8.1}$$

In this work, we propose to leverage advances in model explanations for DNNs and explicitly include a term $\mathcal{L}_{exp}$ that promotes explanation similarity to increase KD fidelity:

$$\mathcal{L} = \mathcal{L}_{KD} + \lambda \mathcal{L}_{exp}. \tag{8.2}$$

Specifically, we maximise the similarity between the models' explanations, for the class $\hat{y}_T$:

$$\mathcal{L}_{exp} = 1 - \mathrm{sim}\left(E(T, x, \hat{y}_T), E(S, x, \hat{y}_T)\right) \ . \tag{8.3}$$

Here, $E(M, x, \hat{y}_T)$ denotes an explanation of model M for class $\hat{y}_T$ and sim a similarity function; in particular, we rely on well-established explanation methods (e.g. GradCAM [SCD+17]) and use cosine similarity in our experiments.

e^{2}KD is Model-agnostic. Note that by computing the loss only across model outputs and explanations, e^{2}KD does not make any reference to architecture-specific details. In contrast to feature distillation approaches, which match specific blocks between teacher and student, e^{2}KD thus holds the potential to seamlessly work across different architectures without any need for adaptation. As we show in Section 8.2, this indeed seems to be the case, with e^{2}KD improving KD performance out of the box for a variety of model architectures, such as CNNs, B-cos CNNs, and even B-cos ViTs [BSFS24].

8.1.2 Evaluating Benefits of e^{2}KD

We now discuss three desiderata that faithful KD should fulfill and why we expect e^{2}KD to be beneficial.

8.1.2.1 *Desideratum 1: High Agreement with the Teacher*

First, faithful KD should ensure that the student classifies any given sample in the same way as the teacher, i.e., it should have high agreement [SIK+21] with the teacher. For inputs $\{x_i\}_{i=1}^N$ this is defined as:

$$\text{Agreement}(T, S) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}_{\hat{y}_{i,T}=\hat{y}_{i,S}} \ . \tag{8.4}$$

While [SIK+21] found that more data points can improve the agreement, in practice, the original dataset that was used to train the teacher might be proprietary or prohibitively large (e.g. [RKH+21]). It is thus highly desirable to effectively distill knowledge *efficiently* with fewer data. As a proxy to evaluate the effectiveness of a given KD approach in such a setting, we propose to use a teacher trained on a large dataset (e.g. ImageNet [DDS+09]) and distill its knowledge to a student using as few as 50 images per class ($\approx 4\%$ of the data) or even perform KD on images of an unrelated dataset.

Compared to standard supervised training, it has been argued that KD improves the student performance by providing more information (full logit distribution instead of binary labels). Similarly, by additionally providing the teachers' explanations, we show that e^{2}KD boosts the performance even further, especially when fewer data is available to learn the same function as the teacher (Section 8.2.1).

8.1.2.2 *Desideratum 2: Learning the 'Right' Features*

Despite achieving high accuracy, models often rely on spurious input features (are not "right for the right reasons" [RHDV17]), and can generalise better if guided to use the 'right' features via human annotations. This is particularly useful in the presence of distribution shifts [SKHL20]. Hence, faithful distillation should ensure that student models also learn to use these 'right' features from a teacher that uses them.

To assess this, we use a binary classification dataset [SKHL20] in which the background is highly correlated with the class label in the training set, making it challenging for models to learn to use the actual class features for classification. We use a teacher that has explicitly been guided to focus on the actual class features and to ignore the background. Then, we evaluate the student's accuracy and agreement with the teacher under distribution shift, i.e., at test time, we evaluate on images in which the class-background correlation is reversed. By providing additional spatial clues from the teachers' explanations to the students, we find that e^{2}KD significantly improves performance over KD (Section 8.2.2).

8.1.2.3 *Desideratum 3: Maintaining Interpretability*

Note that the teacher models might be trained explicitly to exhibit certain desirable properties in their explanations [RBPAS23], or do so as a result of a particular training paradigm [CTM+21] or the model architecture [BSFS24].

We propose two settings to test if such properties are transferred. First, we measure how well the students' explanations reflect properties the teachers were explicitly trained for, i.e. how well they localise class-specific input features when using a teacher that has explicitly been guided to do so [RBPAS23]. We find e^{2}KD to lend itself well to maintaining the interpretability of the teacher, as the explanations of students are explicitly optimised for this (Section 8.2.3.1).

Secondly, we perform a case study to assess whether KD can transfer priors that are not *learnt*, but rather inherent to the model architecture. Specifically, as discussed in Chapter 6, the explanations of B-cos ViTs are sensitive to image shifts, even when shifting by just a few pixels. While we showed that this can be mitigated by using a short convolutional stem in Chapter 6, in Section 8.2.3.2, we find that by learning

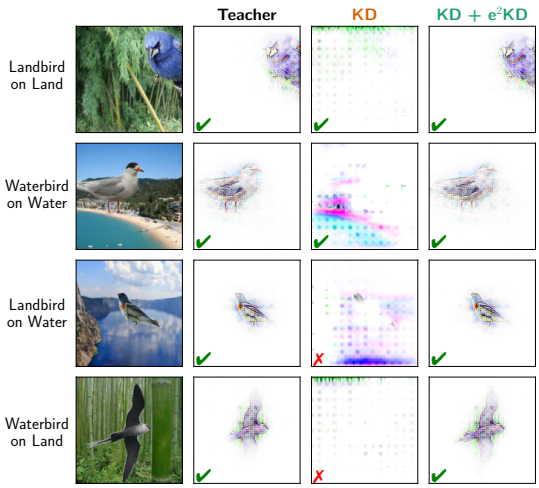


Figure 8.2: **Distillation on biased data.** We distill a B-cos ResNet-50 teacher (*col. 2*) to a B-cos ResNet-18 student with KD (*col. 3*) and e^{2}KD (*col. 4*). While for in-distribution data (*rows 1+2*) the different focus of the models (foreground/background) does not affect the models' predictions (correct predictions marked by ✓), it results in wrong predictions under distribution shift (*rows 3+4*), incorrect predictions marked by ✗).

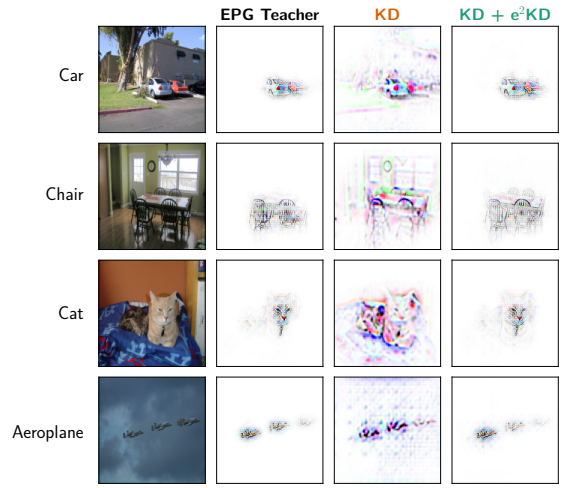


Figure 8.3: **Maintaining focused explanations.** We distill a B-cos ResNet-50 teacher that has been trained with the EPG loss to not focus on confounding input features (*col. 2*), to a B-cos ResNet-18 student with KD (*col. 3*) and e^{2}KD (*col. 4*). The explanations for e^{2}KD are significantly closer to the teacher's (and hence more human-aligned). Samples are from the VOC test set, with all models correctly classifying the shown samples.

from a CNN teacher under e^{2}KD, the explanations of a ViT student without convolutions also become largely invariant to image shifts, and exhibit similar patterns as the teacher.

8.1.3 e^{2}KD with 'Frozen' Explanations

Especially in the 'consistent teaching' setup of [BZR+22], KD requires querying the teacher for every training step, as the input images are repeatedly augmented. To reduce the compute incurred by evaluating the teacher, the idea of using a 'fixed teacher' has been explored by prior logit-based KD approaches [YOH+21, SX22, FPM+23], where logits are pre-computed once at the start of training and used for all augmentations. Analogously, we propose to use pre-computed explanations for images in the e^{2}KD framework. For this, we apply the same augmentations (e.g. cropping or flipping) to images and the teacher's explanations during distillation. In Section 8.2.4, we show that e^{2}KD is robust to such 'frozen' explanations, despite the fact that they of course only approximate the teacher's explanations. As such, frozen explanations provide a trade-off between optimising for explanation similarity and reducing the cost due to the teacher.

8.2 RESULTS

In the following, we present our results. Specifically, in Section 8.2.1 we first compare various KD approaches in terms of accuracy and agreement on the ImageNet dataset as a function of the distillation dataset size. We then present the results on learning the 'right' features from biased data in Section 8.2.2 and on maintaining the interpretability of the teachers in Section 8.2.3. Lastly, in Section 8.2.4, we show that e^{2}KD can also yield significant benefits with approximate 'frozen' explanations (cf. Section 8.1.3).

Before turning to the results, however, in the following we first provide some general details with respect to our training setup and the explanations used for e^{2}KD.

Training Details. In general, we follow the recent KD setup from [BZR+22], which has shown significant improvements for KD; results based on the setup followed by [ZK17, CLZJ21, GYLL23] can be found in Appendix E. Unless specified otherwise, we use the AdamW optimiser [LH19] and, following [BFS22b], do not use weight decay for B-cos models. We use a cosine learning rate schedule with initial warmup for 5 epochs. For the teacher-student logit loss on multi-label VOC dataset, we use the logit loss following [YXZ+23] instead of Equation (8.1). For AT [ZK17], CAT-KD [GYLL23], and ReviewKD [CLZJ21], we follow the original implementation and use cross-entropy based on the ground truth labels instead of Equation (8.1); for an adaptation to B-cos models, see Appendix E. For each method and setting, we report the results of the best hyperparameters (softmax temperature and the methods' loss coefficients) as obtained on a separate validation set. Unless specified otherwise, we augment images via random horizontal flipping and random cropping with a final resize to 224×224. For full details, see Appendix E.

Explanation Methods. For e^{2}KD, we use GradCAM [SCD+17] for standard models and B-cos explanations for B-cos models, optimising the cosine similarity as per Equation (8.3). For B-cos, we use the dynamic weights $\mathbf{W}(\mathbf{x})$ as explanations [BFS22b].

8.2.1 e^{2}KD Improves Learning from Limited Data

Setup. To test the robustness of e^{2}KD with respect to the dataset size (Section 8.1.2.1), we distill with 50 ($\approx 4\%$) or 200 ($\approx 16\%$) shots per class, as well as the full ImageNet training data; further, we also test without any access to ImageNet, by performing KD on the SUN397 [XHE+10], whilst still evaluating on ImageNet (and vice versa). We distill ResNet-34 [HZRS16] teachers into ResNet-18 students, both for standard and B-cos models; additionally, we use a B-cos DenseNet-169 [HLVDMW17] as a teacher to evaluate distillation across architectures. For reference, we also provide results as obtained via attention transfer (AT) [ZK17], CAT-KD [GYLL23], and ReviewKD [CLZJ21].

Results. In Tables 8.1 and 8.2, we show that e^{2}KD can significantly improve KD in terms of top-1 accuracy as well as top-1 agreement with the teacher on ImageNet. We observe particularly large gains for small distillation dataset sizes. E.g., accuracy and agreement for conventional models on 50 shots improve by 5.1 (B-cos: 8.6) and 6.2 (B-cos: 10.0) p.p. respectively. In Table 8.5, we show that e^{2}KD also provides significant gains in the 'data-free' setting [BZR+22], improving the accuracy and agreement of the student by 4.9 and 5.4 p.p. respectively, despite computing the explanations on an unrelated dataset (i.e. SUN$\rightarrow$ImageNet, right). Similar gains can be observed when using ImageNet images to distill a teacher trained on SUN (i.e. ImageNet $\rightarrow$SUN, left).

8.2.2 e^{2}KD Improves Learning the 'Right' Features

Setup. To assess whether the students learn to use the same input features as the teacher (Section 8.1.2.2), we use the Waterbirds-100 dataset [SKHL20], a binary classification task between land- and waterbirds, in which birds are highly correlated with the image backgrounds during training. As teachers, we use pre-trained ResNet-50 models from [RBPAS23], which were guided to use the bird features instead of the background; as in Section 8.2.1, we use conventional and B-cos models and provide results obtained via prior work for reference.

Standard Models Teacher ResNet-34 Accuracy 73.3%	**50 Shots**		**200 Shots**		**Full data**	
	Accuracy	Agreement	Accuracy	Agreement	Accuracy	Agreement
Baseline ResNet-18	23.3	24.8	47.0	50.2	69.8	76.8
AT [ZK17]	38.3	41.1	54.7	59.0	69.7	74.9
ReviewKD [CLZJ21]	51.2	55.6	63.0	69.0	71.4	80.0
CAT-KD [GYLL23]	32.1	34.5	52.4	56.7	70.9	78.7
KD [HVD15, BZR+22]	49.8	55.5	63.1	71.9	**71.8**	81.2
+ e^{2}KD (GradCAM)	**54.9** (+ 5.1)	**61.7** (+ 6.2)	**64.1** (+ 1.0)	**73.2** (+ 1.3)	**71.8** (+ 0.0)	**81.6** (+ 0.4)

Table 8.1: **KD on ImageNet for standard models.** For a ResNet-34 teacher and a ResNet-18 student, we show the accuracy and agreement of various KD approaches for three different distillation dataset sizes. We observe that across all settings e^{2}KD yields significant top-1 accuracy gains over vanilla KD, while remaining competitive with prior work. Crucially, e^{2}KD also exhibits the highest degree of distillation fidelity as measured by the top-1 agreement with the teacher. Similar results are also observed for B-cos models, see Table 8.2.

B-cos Models Teacher ResNet-34 Accuracy 72.3%	**50 Shots**		**200 Shots**		**Full data**	
	Accuracy	Agreement	Accuracy	Agreement	Accuracy	Agreement
Baseline ResNet-18	32.6	35.1	53.9	59.4	68.7	76.9
AT [ZK17]	32.6	35.8	45.8	51.1	69.0	77.2
ReviewKD [CLZJ21]	47.5	53.2	54.1	60.8	57.0	64.6
CAT-KD [GYLL23]	53.1	59.8	58.6	66.4	63.9	73.7
KD [HVD15, BZR+22]	35.3	38.4	56.5	62.9	70.3	79.9
+ e^{2}KD (B-cos)	**43.9** (+ 8.6)	**48.4** (+10.0)	**58.8** (+ 2.3)	**66.0** (+ 3.1)	**70.6** (+ 0.3)	**80.3** (+ 0.4)

Table 8.2: **KD on ImageNet for B-cos models.** For a B-cos ResNet-34 teacher and a B-cos ResNet-18 student, we show the accuracy and agreement of various KD approaches for three different distillation dataset sizes. We observe that across all settings e^{2}KD yields significant accuracy and agreement gains over vanilla KD, whilst also remaining competitive with prior work.

B-cos Models Teacher DenseNet-169 Accuracy 75.2%	**50 Shots**		**200 Shots**		**Full data**	
	Accuracy	Agreement	Accuracy	Agreement	Accuracy	Agreement
Baseline ResNet-18	32.6	34.5	53.9	58.4	68.7	75.5
KD [HVD15, BZR+22]	37.3	40.2	51.3	55.6	71.2	78.8
+ e^{2}KD (B-cos)	**45.4** (+ 8.1)	**49.0** (+ 8.8)	**55.7** (+ 4.4)	**60.7** (+ 5.1)	**71.9** (+ 0.7)	**79.8** (+ 1.0)
❄ KD	33.4	35.7	50.4	54.5	68.7	75.2
❄ **+ e^{2}KD (B-cos)**	**38.7** (+ 5.3)	**41.7** (+ 6.0)	**53.6** (+ 3.2)	**58.3** (+ 3.8)	**69.5** (+ 0.8)	**76.4** (+ 1.2)

Table 8.3: **KD on ImageNet for B-cos models for a DenseNet-169 teacher and with 'frozen' (❄) explanations and logits.** Similar to the results in Table 8.2, we find that e^{2}KD adds significant gains to 'vanilla' KD across dataset sizes (50 Shots, 200 Shots, full data). Given that e^{2}KD does not rely on matching specific blocks between architectures (cf. [CLZJ21, ZK17]), it seamlessly generalises across architectures. Further, e^{2}KD can also be performed with 'frozen' (❄) explanations by augmenting images and pre-computed explanations jointly (Section 8.1.3).

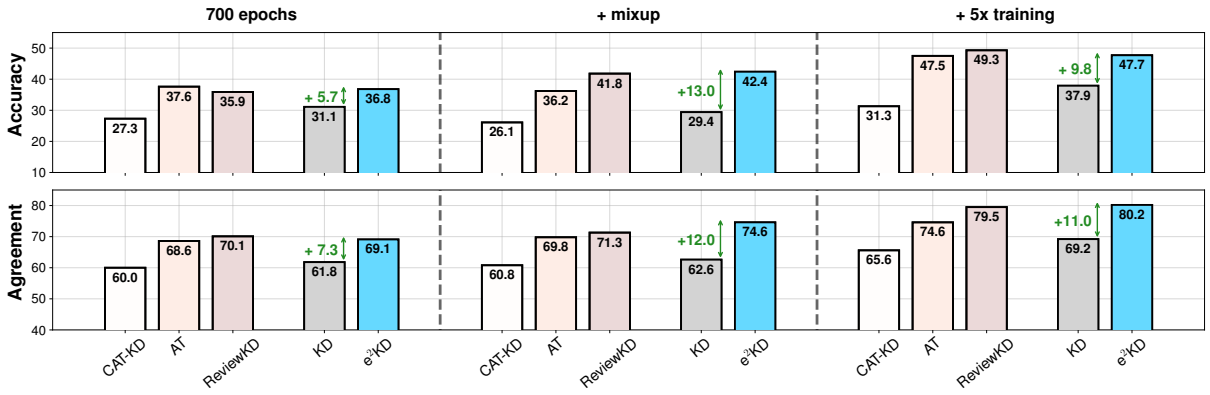


Figure 8.4: **KD on out-of-distribution data for standard models on Waterbirds-100.** Accuracy (*top*) and agreement (*bottom*) when distilling from a ResNet-50 teacher to a ResNet-18 student with various KD approaches. Following [BZR+22], we additionally evaluate the effectiveness of adding mixup (*col. 2*) and, additionally, long teaching (*col. 3*). We find that our proposed e^{2}KD provides significant benefits in both accuracy and agreement over vanilla KD, and is further enhanced by the use of long teaching and mixup. We show the performance of prior work for reference, and find that e^{2}KD performs competitively. For results on B-cos models, please see Appendix E.

Results. In Figure 8.4, we present our results on the Waterbirds dataset for standard models. Specifically, we evaluate the accuracy and student-teacher agreement of each method on object-background combinations not seen during training (i.e. 'Waterbirds on Land' and 'Landbirds on Water') to assess how well the students learnt from the teacher to use the 'right' features (i.e. the birds).

In light of the findings by [BZR+22] that long teaching schedules and strong data augmentations help, we do so for three settings[1]: (1) for 700 epochs, (2) with additional mixup [ZCDLP18] for data augmentation, as well as (3) for training 5x longer ('patient teaching').

Across settings, we find that e^{2}KD consistently and significantly improves the performance of KD both in terms of accuracy and agreement. Despite its simplicity, it compares favourably to prior work in terms of accuracy and agreement, indicating that e^{2}KD indeed helps improving KD fidelity. For quantitative results obtained with B-cos models, see Figure 8.1 (centre) and Appendix E. We also find clear qualitative improvements in the model's ability to focus on the 'right' features, see Figure 8.2.

Further, consistent with [BZR+22], we find mixup augmentation and longer training schedules to also significantly improve agreement. This provides additional evidence for the hypothesis put forward by [BZR+22] that KD *could* be sufficient for function matching if performed for long enough. As such, and given the simplicity of the dataset, the low resource requirements, and a clear target (100% agreement on unseen combinations), we believe the Waterbirds dataset to constitute a great benchmark for future research on KD fidelity.

8.2.3 e^{2}KD Improves the Student's Interpretability

In this section, we show results on maintaining the teacher's interpretability (cf. Section 8.1.2.3). In particular, we show that e^{2}KD naturally lends itself to distilling localisation properties into the student that the teacher was trained for (Section 8.2.3.1) and that even architectural priors of a CNN teacher can be transferred to a ViT student (Section 8.2.3.2).

[1]Compared to ImageNet, the small size of the Waterbirds-100 dataset allows for reproducing the 'patient teaching' results with limited compute.

	EPG Teacher			IoU Teacher		
	EPG	IoU	F1	EPG	**IoU**	F1
Teacher ResNet-50	75.7	21.3	72.5	65.0	49.7	72.8
Baseline ResNet-18	50.0	29.0	58.0	50.0	29.0	58.0
KD [YXZ+23]	60.1	31.6	60.1	58.9	35.7	62.7
+ e²KD (B-cos)	**71.1**	24.8	**67.6**	60.3	**45.7**	**64.8**

Table 8.4: **e²KD on VOC.** We compare KD and e²KD when distilling from a teacher guided [RBPAS23] to either optimise for EPG (*left*) or IoU (*right*). Explanations of the e²KD student better align with those of the teacher, as evidenced by significantly higher EPG (IoU) scores when distilled from the EPG (IoU) teacher. e²KD students also achieve higher accuracy (F1).

	IMN → SUN		SUN → IMN	
	Acc.	Agr.	Acc.	Agr.
Teacher DenseNet-169	60.5	-	75.2	-
Baseline ResNet-18	57.7	67.9	68.7	75.5
KD [HVD15]	53.5	65.0	14.9	16.7
+ e²KD (B-cos)	**54.9**	**67.7**	**19.8**	**22.1**

Table 8.5: **'Data-free' KD**. We distill a B-cos DenseNet-169 teacher model, *left*: trained on the SUN [XHE+10] dataset using ImageNet (IMN→SUN), and *right:* trained on ImageNet using SUN (SUN→IMN). In both cases, we see that the B-cos ResNet-18 student distilled with e²KD achieves significantly higher accuracy and agreement scores than student trained via vanilla KD.

8.2.3.1 *Maintaining Focused Explanations*

Setup. To assess whether the students learn to give similar explanations as the teachers, we distill B-cos ResNet-50 teachers into B-cos ResNet-18 students on PASCAL VOC 2007 [EVGW+09] in a multi-label classification setting. Specifically, we use two different teachers: one with a high EPG (EPG Teacher), and one with a high IoU score (IoU Teacher). To quantify the students' focus, we measure the EPG and IoU scores [RBPAS23] of the explanations with respect to the dataset's bounding box annotations in a multi-label classification setting. As these teachers are trained explicitly to exhibit certain properties in their explanations, a *faithfully distilled* student should optimally exhibit the same properties.

Results. As we show in Table 8.4, the explanations of an e²KD student indeed more closely mirror those of the teacher than a student trained via vanilla KD: e²KD students exhibit significantly higher EPG when distilled from the EPG teacher (EPG: 71.1 vs. 60.3) and vice versa (IoU: 45.7 vs. 24.8). In contrast, 'vanilla' KD students show only minor differences (EPG: 60.1 vs. 58.9; IoU: 35.7 vs. 31.6). Figure 8.3 shows that these improvements are also reflected qualitatively, with the e²KD students reflecting the teacher's focus much more faithfully in their explanations.

While this might be expected as e²KD explicitly optimises for explanation similarity, we would like to highlight that this not only ensures that the desired properties of the teachers are better represented in the student model, but also significantly improves the students' performance (e.g., F1: 60.1→67.6 for the EPG teacher). As such, we find e²KD to be an easy-to-use and effective addition to vanilla KD for improving both interpretability as well as task performance.

8.2.3.2 *Distilling Architectural Priors*

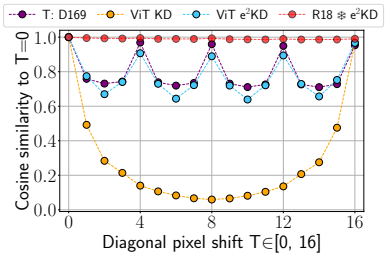
Setup. To assess whether students can learn architectural priors of the models, we evaluate whether a B-cos ViT_{Tiny} student model can learn to give explanations that are similar to those of a pretrained CNN (B-cos DenseNet-169) teacher model; for this, we again use the ImageNet dataset.

Results. In line with the preceding results, we find (Figure 8.5, left) that e²KD significantly improves the accuracy of the ViT student model (64.8→66.3), as well as the agreement with the teacher (70.1→71.8).

Interestingly, we find that the ViT student's explanations seem to become similarly robust to image shifts as those of the teacher (Figure 8.5, centre and right). Specifically, note that the image tokenisation of the ViT model using vanilla KD (extracting non-overlapping patches of size 16×16) induces a periodicity of

KD from CNN to ViT

Method	Acc.	Agr.
T: CNN	75.2	-
B: ViT	60.0	64.6
KD	64.8	70.1
+ e^{2}KD (B-cos)	**66.3**	**71.8**



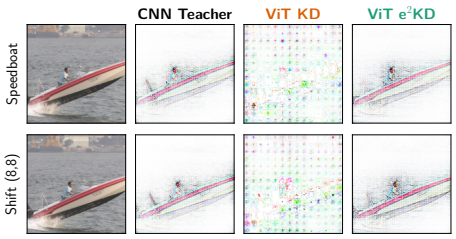


Figure 8.5: **Distilling inductive biases from CNN to ViT.** We distill a B-cos DenseNet-169 teacher to a B-cos ViT$_{\text{Tiny}}$. **Left:** We find e^{2}KD to yield significant gains in both accuracy and agreement. **Centre:** Cosine similarity of explanations for shifted images w.r.t. the unshifted image (T=0). We find that under e^{2}KD (blue) the ViT student learns to mimic the shift periodicity of the teacher (purple), despite the inherent periodicity of 16 of the ViT architecture (seen for vanilla KD, yellow). Notably, e^{2}KD with frozen explanations yields shift-equivariant students (red), see also Section 8.2.3.2. **Right:** e^{2}KD significantly improves the explanations of the ViT model, thus maintaining the utility of the explanations of the teacher model. While the explanations for KD change significantly under shift (subcol. 3), for e^{2}KD (subcol. 4), as with the CNN teacher (subcol. 2), the explanations remain consistent. For more qualitative samples, see Appendix E.

16 with respect to image shifts T, see, e.g., Figure 8.5 (centre, yellow curve): here, we plot the cosine similarity of the explanations[2] at various shifts with respect to the explanation given for the original, unshifted image (T=0). In contrast, due to smaller strides (stride$\in\{1,2\}$ for any layer) and overlapping convolutional kernels, the CNN teacher model is inherently more robust to image shifts, see Figure 8.5 (centre, purple curve), exhibiting a periodicity of 4. A ViT student trained via e^{2}KD learns to mimic the teacher (see Figure 8.5, centre, blue curve) and exhibits the same periodicity, indicating that e^{2}KD indeed helps the student learn a function more similar to the teacher. Importantly, this also helps improving the explanations of the ViT model, see Figure 8.5 (right). In particular, we show the explanations for images at a diagonal shift of 0 and 8 pixels. As can be seen, the explanations become more robust to such shifts and more easily interpretable, thus maintaining the utility of the explanations of the teacher.

Notably, our results indicate that it might be possible to instill desired properties into a DNN model even beyond knowledge distillation. E.g., note that the frozen explanations (see also Section 8.2.4) do not exhibit the CNN's periodicity, as *by design* they maintain consistent augmentations across shifts and crops. Based on our observations for the ViTs, we thus expect that a student trained on frozen explanations becomes *fully shift-equivariant*, which is indeed the case (see Figure 8.5, centre, red curve, ResNet-18).

8.2.4 e^{2}KD with Frozen Explanations

In the previous sections, we have seen that e^{2}KD is a robust approach that provides consistent gains even when only limited data is available (see Section 8.2.1) and works across different architectures (e.g., DenseNet→ResNet or DenseNet→ViT, see Sections 8.2.1 and 8.2.3.2). In the following, we show that e^{2}KD even works when only 'approximate' explanations for the teacher are available (cf. Section 8.1.3).

Setup. To test the robustness of e^{2}KD when using frozen explanations, we distill from a B-cos DenseNet-169 teacher to a B-cos ResNet-18 student using pre-computed, frozen explanations on the ImageNet dataset. We also evaluate across varying dataset sizes, as in Section 8.2.1.

Results. Table 8.3 (bottom) shows that e^{2}KD with frozen explanations is effective for improving both the

[2]We compute the similarity of the intersecting area of the explanations.

accuracy and agreement over KD with frozen logits across dataset sizes (e.g. accuracy: 33.4→38.7 for 50 shots). Furthermore, e^2KD with frozen explanations also outperforms vanilla KD under both metrics when using limited data (e.g. accuracy: 37.3→38.7 for 50 shots). As such, a frozen teacher constitutes a more cost-effective alternative for obtaining the benefits of e^2KD, whilst also highlighting its robustness to using 'approximate' explanations.

8.3 CONCLUSION

We proposed a simple approach to improve knowledge distillation (KD) by explicitly optimising for the explanation similarity between the teacher and the student, and showed its effectiveness in distilling the teacher's properties under multiple settings. Specifically, e^2KD helps the student (1) achieve competitive and often higher accuracy and agreement than vanilla KD, (2) learn to be 'right for the right reasons', and (3) learn to give similar explanations as the teacher, e.g. even when distilling from a CNN teacher to a ViT student. Finally, we showed that e^2KD is robust in the presence of limited data, approximate explanations, and across model architectures. In short, we find e^2KD to be a simple but versatile addition to KD that allows for more faithfully distilling a teacher's knowledge into a student while also maintaining competitive task performance.

IV

CONTRASTIVE LEARNING ON LONG-TAIL DATA

For the last chapter of this thesis, we switch gears and turn our attention to a different problem: instead of studying (Parts I and II) and improving (Part III) individual model decisions via explanations, we aim to better understand a commonly used framework for self-supervised representation learning.

In particular, in **Chapter 9**, we study the effect of a single hyperparameter on the learning dynamics of contrastive learning methods: the 'softmax temperature' τ. We show that this parameter critically influences the learnt representations and that periodically changing it throughout the training process yields significant benefits on long-tail datasets.

9 Improving Representations via Temperature Schedules

Contents

Deep Neural Networks (DNNs) excel at learning data representations that are useful for a variety of tasks. Especially since the advent of recent self-supervised learning (SSL) techniques, rapid progress towards learning universally useful representations has been made.

Currently, however, SSL on images is mainly carried out on datasets that have been constructed and curated for supervised learning (e.g. ImageNet [DDS+09], CIFAR [Kri09], etc.). Although the labels of curated datasets are not *explicitly* used in SSL, the class-balanced *structure* of the data could still result in a learning signal for unsupervised methods. As such, these methods are often not evaluated in the settings they were designed for, i.e. learning from truly unlabelled data. In fact, some methods (e.g. [ARV19, CMM+20]) even explicitly enforce a uniform prior over the embedding or label space, which cannot be expected to hold for uncurated datasets.

Since uncurated, real-world data tends to follow long-tail distributions [Ree01], in this chapter, we analyse SSL methods on long-tailed data. Specifically, we analyse the behaviour of contrastive learning (CL) methods, which are among the most popular learning paradigms for SSL.

In CL, the models are trained such that embeddings of different samples are repelled, while embeddings of different 'views'of the same sample are attracted. The strength of those attractive and repelling forces is controlled by a temperature parameter τ, which has been shown to play a crucial role in learning good representations [CFGH20, CKNH20]. To the best of our knowledge, τ has thus far almost exclusively been treated as a *constant* hyperparameter. In contrast, we employ a *dynamic* τ during training and show that introducing a simple schedule for τ consistently improves the representation quality across a wide range of long-tail distribution settings. Crucially, these gains come without additional costs and only require oscillating τ with a cosine schedule.

This mechanism is grounded in our novel understanding of the effect of temperature on the contrastive loss. In particular, we analyse the contrastive loss from an average distance maximisation perspective, which gives intuitive insights as to why a large temperature emphasises *group-wise discrimination*, whereas a small temperature leads to a higher degree of *instance discrimination* and more uniform distributions over the embedding space. Varying τ during training ensures that the model learns both group-wise and instance-specific features, resulting in better separation between head and tail classes.

Overall, our contributions are summarised as follows:

(1) We carry out an extensive analysis of the effect of τ on imbalanced data,

(2) analyse CL from an average distance perspective to understand the emergence of semantic structure,

(3) propose a simple yet effective schedule for τ that improves performance across various settings,

(4) and show that the proposed τ scheduling is robust with respect to different hyperparameter choices.

This chapter is based on [KBS+23] and the corresponding code is publicly available at:
github.com/Annusha/temperature_schedules.

9.1 METHOD

In the following, we describe our approach and analysis of contrastive learning on long-tailed data. For this, we will first review the core principles of contrastive learning for the case of uniform data (Section 9.1.1). In Section 9.1.2, we then place a particular focus on the temperature parameter τ in the contrastive loss and its impact on the learnt representations. Based on our analysis, in Section 9.1.3 we discuss how the choice of τ might negatively affect the learnt representation of rare classes in the case of long-tailed distributions. Following this, we describe a simple proof-of-concept based on additional coarse supervision to test our hypothesis. We then further develop temperature schedules (TS) that yield significant gains with respect to the separability of the learnt representations in Section 9.2.

9.1.1 Contrastive Learning

The Info-NCE loss is a popular objective for contrastive learning (CL) and has lead to impressive results for learning useful representations from unlabelled data [OLV18, WXSL18, HFW+20, CKNH20]. Given a set of inputs $\{x_1, \dots, x_N\}$, and the cosine similarities s_{ij} between learnt representations $u_i = f(\mathcal{A}(x_i))$ and $v_j = g(\mathcal{A}(x_j))$ of the inputs, the loss is defined by:

$$\mathcal{L}_{\text{c}} = \sum_{i=1}^{N} -\log \frac{\exp\left(s_{ii}/\tau\right)}{\exp\left(s_{ii}/\tau\right) + \sum_{j \neq i} \exp\left(s_{ij}/\tau\right)}. \tag{9.1}$$

Here, $\mathcal{A}(\cdot)$ applies a random augmentation to its input and f and g are deep neural networks. For a given x_i, we will refer to u_i as the *anchor* and to v_j as a *positive* sample if $i = j$ and as a *negative* if $i \neq j$. Last, τ denotes the *temperature* of the Info-NCE loss and has been found to crucially impact the learnt representations of the model [WI20, WL21a, RSY+21].

Uniformity. Specifically, a small τ has been tied to more uniformly distributed representations, see Figure 9.1. For example, [WL21a] show that the loss is 'hardness-aware', i.e. negative samples closest to the anchor receive the highest gradient. In particular, for a given anchor, the gradient with respect to the negative sample v_j is scaled by its relative contribution to the denominator in Equation (9.1):

$$\frac{\partial \mathcal{L}_c}{\partial v_j} = \frac{\partial \mathcal{L}_c}{\partial s_{ij}} \times \frac{\partial s_{ij}}{\partial v_j} = \frac{1}{\tau} \times [\text{softmax}_k(s_{ik}/\tau)]_j \times \frac{\partial s_{ij}}{\partial v_j} \quad . \tag{9.2}$$

As a result, for sufficiently small τ, the model minimises the cosine similarity to the nearest negatives in the embedding space, as softmax approaches an indicator function that selects the largest gradient. The optimum of this objective, in turn, is to distribute the embeddings as uniformly as possible over the sphere, as this reduces the average similarity between nearest neighbours, see also Figures 9.1 and 9.3.

Semantic Structure. In contrast, a large τ has been observed to induce more semantic structure in the representation space. However, while the effect of small τ has an intuitive explanation, the phenomenon that larger τ induce semantic structure is much more poorly understood and has mostly been described empirically [WL21a, RSY+21]. Specifically, note that for any given positive sample, all negatives are repelled from the anchor, with close-by samples receiving exponentially higher gradients. Nonetheless, for large τ, tightly packed semantic clusters emerge. However, if close-by negatives are heavily repelled, how can this be? Should the loss not be dominated by the hard-negative samples and thus break the semantic structure?

To better understand both phenomena, we propose to view the contrastive loss through the lens of *average distance* maximisation, which we describe in the following section.

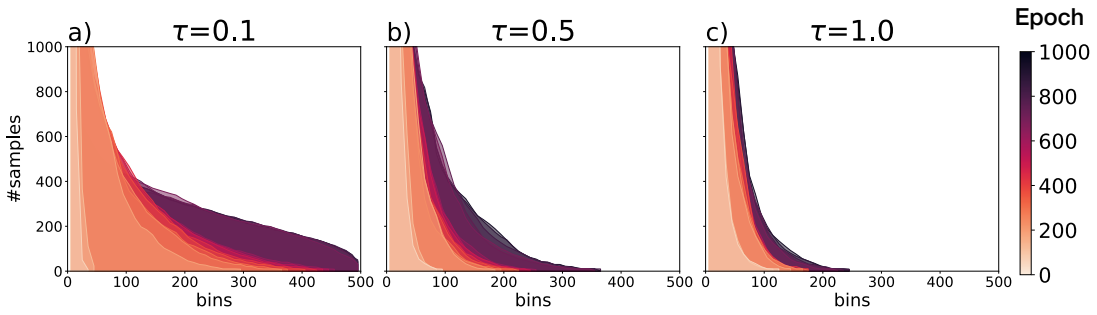


Figure 9.1: **Coverage of the embedding space during training.** To measure coverage we uniformly sample 500 bins on the unit sphere. Each training sample is assigned to the closest bin and we plot a histogram of the assignments. X-axis: bins. Y-axis: number of training samples in a bin. Colors denotes epochs: light is the 1st epoch of training, dark is the last. For small τ (a) the representations are more uniformly distributed (cf. Section 9.1).

9.1.2 Contrastive Learning as Average Distance Maximisation

As discussed in the previous section, the parameter τ plays a crucial role in shaping the learning dynamics of contrastive learning. To understand this role better, in this section, we present a novel viewpoint on the mechanics of the contrastive loss that explain the observed model behaviour. In particular, and in contrast to [WL21a] who focused on the impact of *individual* negatives, for this we discuss the *cumulative* impact that all negative samples have on the loss.

To do so, we express the summands $\mathcal{L}_c^i$ of the loss in terms of distances d_{ij} instead of similarities s_{ij}:

$$0 \;\leq\; d_{ij} \;=\; \frac{1 - s_{ij}}{\tau} \;\leq\; \frac{2}{\tau} \quad \text{and} \quad c_{ii} = \exp(d_{ii}). \tag{9.3}$$

This allows us to rewrite the loss $\mathcal{L}_c^i$ as

$$\mathcal{L}_{\mathrm{c}}^i = -\log\left(\frac{\exp\left(-d_{ii}\right)}{\exp\left(-d_{ii}\right) + \sum_{j\neq i}\exp\left(-d_{ij}\right)}\right) = \log\left(1 + c_{ii}\sum_{j\neq i}\exp\left(-d_{ij}\right)\right). \tag{9.4}$$

As the effect c_{ii} of the positive sample for a given anchor is the same for all negatives, in the following we place a particular focus on the negatives and their relative influence on the loss in Equation (9.4); for a discussion of the influence of positive samples, please see Section F.4 in the appendix. To understand the impact of the temperature τ, first note that the loss monotonically increases with the sum $S_i = \sum_{j\neq i}\exp(-d_{ij})$ of exponential distances in Equation (9.4). As $\log$ is a continuous, monotonic function, we base the following discussion on the impact of τ on the sum S_i.

For Small τ, the nearest neighbours of the anchor point dominate S_i, as differences in similarity are amplified. As a result, the contrastive objective maximises the average distance to nearest neighbours, leading to a uniform distribution over the hypersphere, see Figure 9.3. Since individual negatives dominate the loss, this argument is consistent with existing interpretations, e.g. [WL21a], as described in the previous section.

For Large τ, (e.g. $\tau \geq 1$), on the other hand, the contributions to the loss from a given negative are on the same order of magnitude for a wide range of cosine similarities. Hence, the constrastive objective can be thought of as maximising the average distance over a wider range of neighbours. Interestingly, since distant negatives will typically outnumber close negatives, the strongest *cumulative* contribution to the contrastive loss will come from more distant samples, despite the fact that *individually* the strongest contributions will

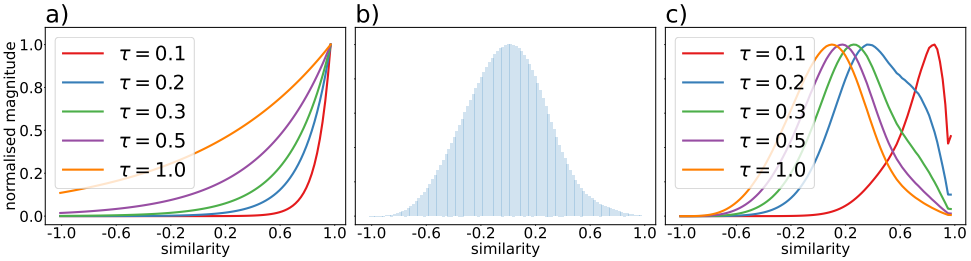


Figure 9.2: **Loss contribution by similarity.** X-axis: cosine similarity between anchor and negative. All curves are normalised such that their max y-value is 1. **a)**: influence of an individual negative sample to the loss depending on its similarity to anchor for different τ; **b)**: average histogram of distribution of negatives over the hypersphere with respect to their similarity to the anchor; **c)**: cumulative impact that negative samples have on the loss. The *cumulative* contribution of negatives shifts left, towards less similar samples, in contrast to individual contributions of negatives. As $\tau \to \infty$, the cumulative distribution coincides with the histogram b).

come from the closest samples. To visualise this, in Figure 9.2a, we plot the contributions of *individual* samples depending on their distance, as well as the distribution of similarities s_{ij} to negatives over the entire dataset in Figure 9.2b. Since the number of negatives at larger distances (e.g. $s_{ij} \approx 0.1$) significantly outnumber close negatives ($s_{ij} > 0.9$), the peak of the cumulative contributions[1] shifts towards lower similarities for larger τ, as can be seen in Figure 9.2c; in fact, for $\tau \to \infty$, the distribution of cumulative contributions approaches the distribution of negatives.

Hence, the model can significantly decrease the loss by increasing the distance to relatively 'easy negatives' for much longer during training, i.e. to samples that are easily distinguishable from the anchor by simple patterns. Instead of learning 'hard' features that allow for better *instance discrimination* between hard negatives, the model will be biased to learn easy patterns that allow for *group-wise discrimination* and thereby increase the margin between clusters of samples. Note that since the clusters as a whole mutually repel each other, the model is optimised to find a trade-off between the expanding forces between hard negatives (i.e. within a cluster) and the compressing forces that arise due to the margin maximisation between easy negatives (i.e. between clusters).

Importantly, such a bias towards easy features can prevent the models from learning hard features—i.e. by focusing on *group-wise discrimination*, the model becomes agnostic to instance-specific features that would allow for a better *instance discrimination* (cf. [RSY+21]). In the following, we discuss how this might negatively impact rare classes in long-tailed distributions.

9.1.3 Temperature Schedules for Contrastive Learning on Long-tail Data

As discussed in the introduction to this chapter, naturally occurring data typically exhibit long-tail distributions, with some classes occurring much more frequently than others; across the dataset, *head* classes appear frequently, whereas *tail* classes contain fewest number of samples. Since self-supervised learning methods are designed to learn representations from unlabelled data, it is important to investigate their performance on imbalanced datasets.

Claim: Tail Classes Benefit from Instance Discrimination. As discussed in Section 9.1.2, sufficiently large τ are required for semantic groups to emerge during contrastive learning as this emphasises group-wise discrimination. However, as shown by [RSY+21], this can come at the cost of encoding

[1] To obtain the cumulative contributions, we group the negatives into 100 non-overlapping bins of size 0.02 depending on their distance to the anchor and report the sum of contributions of a given bin.

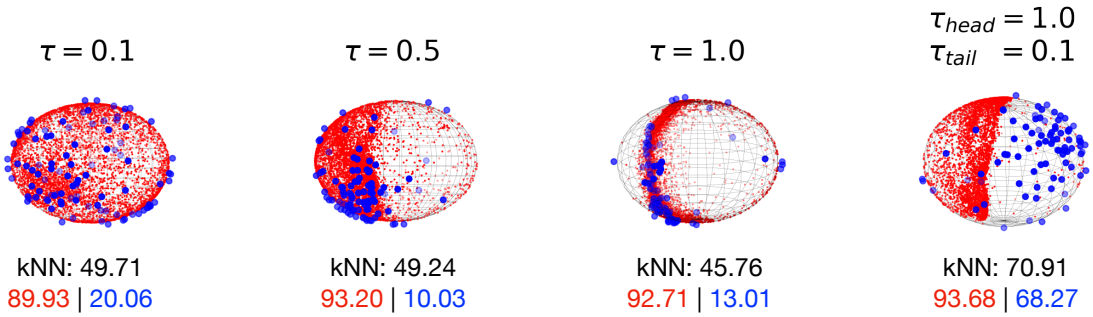


Figure 9.3: **Representations of a head and a tail class.** Visualisation of the influence of τ on representations of two semantically close classes (trained with all 10 classes). Red: single head class and blue: single tail class from CIFAR10-LT. Small $\tau{=}0.1$ promotes uniformity, while large $\tau{=}1.0$ creates dense clusters. With $\tau_{\{head/tail\}}$ we refer to coarse supervision described in Section 9.1.3 which separates tail from head classes. In black / red / blue, we respectively show the average kNN accuracy over all classes / the head class / the tail class.

instance-specific features and thus hurt the models' instance discrimination capabilities.

We hypothesise that this disproportionately affects tail classes, as tail classes consist of only relatively few instances to begin with. Their representations should thus *remain distinguishable* from most of their neighbours and not be grouped with other instances, which are likely of a different class. In contrast, since head classes are represented by many samples, grouping those will be advantageous.

To test this hypothesis, we propose to explicitly train head and tail classes with different τ, to emphasise group discrimination for the former while ensuring instance discrimination for the latter.

Experiment: Controlling τ with Coarse Supervision. We experiment on CIFAR10-LT (a long-tail variant of CIFAR10- see Section 9.2.1) in which we select a different τ depending on whether the anchor u_i is from a head or a tail class, i.e. of the 5 *most* or *least* common classes. We chose a relatively large τ ($\tau_{\text{head}}{=}1.0$) for the 5 head classes to emphasise group-wise discrimination and a relatively small τ ($\tau_{\text{tail}}{=}0.1$) for the 5 tail classes to encourage the model to learn instance-discriminating features.

As can be seen in Figure 9.3, this simple manipulation of the contrastive loss indeed provides a significant benefit with respect to the semantic structure of the embedding space, despite only weakly supervising the learning by adjusting τ according to a coarse (frequent/infrequent) measure of class frequency.

In particular, in Figure 9.3, we show the projections of a single head class and a single tail class onto the three leading PCA dimensions and the corresponding kNN accuracies. We would like to highlight the following results. First, without any supervision, we indeed find that the head class consistently performs better for larger values of τ (e.g. 1.0), whereas the tail class consistently benefits from smaller values for τ (e.g. 0.1). Second, when training the model according to the coarse τ supervision as described above, we are not only able to maintain the benefits of large τ values for the head class, but significantly outperform all constant τ versions for the tail class, which improves the overall model performance on all classes; detailed results for all classes are provided in Appendix F.

Temperature Schedules (TS) without supervision. Such supervision with respect to the class frequency is, of course, generally not available when training on unlabelled data and these experiments are only designed to test the above claim and provide an intuition about the learning dynamics on long-tail data. However, we would like to point out that the supervision in these experiments is very coarse and only separates the unlabelled data into *frequent* and *infrequent* classes. Nonetheless, while the results are encouraging, they are, of course, based on additional, albeit coarse, labels. Therefore, in what follows, we

present an unsupervised method that yields similar benefits.

In detail, we propose to modify τ according to a cosine schedule, such that it alternates between an upper (τ_+) and a lower (τ_-) bound at a fixed period length T:

$$\tau_{\cos}(t) = (\tau_+ - \tau_-) \times (1 + \cos(2\pi\, t/T))/2 + \tau_- \; ; \tag{9.5}$$

here, t denotes training epochs. This method is motivated by the observation that τ controls the trade-off between learning easily separable features and learning instance-specific features.

Arguably, however, the models should learn both types of features: i.e. the representation space should be structured according to easily separable features that (optimally) represent semantically meaningful group-wise patterns, whilst still allowing for instance discrimination within those groups.

Therefore, we propose to *alternate* between both objectives as in Equation (9.5), to ensure that throughout training the model learns to encode instance-specific patterns, whilst also structuring the representation space along semantically meaningful features. Note that while we find a cosine schedule to work best and to be robust with respect to the choice for T (Section 9.2.3), we also evaluate alternatives. Even randomly sampling τ from the interval $[\tau_-, \tau_+]$ improves the model performance. This indicates that the *task switching* between group-wise discrimination (large τ) and instance discrimination (small τ) is indeed the driving factor behind the performance improvements we observe.

9.2 EXPERIMENTAL RESULTS

In this section, we validate our hypothesis that simple manipulations of the temperature τ in Equation (9.1) lead to better performance for long-tailed data. First, we introduce our experimental setup in Section 9.2.1, then in Section 9.2.2 we discuss the results across three imbalanced datasets and, finally, we analyse different design choices of the framework through extensive ablation studies in Section 9.2.3.

9.2.1 Implementation Details

Datasets. We consider long-tailed (LT) versions of the following three popular datasets for the experiments: CIFAR10-LT, CIFAR100-LT, and ImageNet 100-LT. For most of the experiments, we follow the setting from SDCLR [JCMW21]. In case of **CIFAR10-LT/CIFAR100-LT**, the original datasets [Kri09] consist of 60000 32x32 images sampled uniformly from 10 and 100 semantic classes, respectively, where 50000 images correspond to the training set and 10000 to a test set. Long-tail versions of the datasets are introduced by [CJL+19] and consist of a subset of the original datasets with an exponential decay in the number of images per class. The imbalance ratio controls the uniformity of the dataset and is calculated as the ratio of the sizes of the biggest and the smallest classes. By default, we use an imbalance ratio 100 if not stated otherwise. Experiments in Table 9.1, Table 9.3 are the average over three runs with different permutations of classes. **ImageNet100-LT** is a subset of the original ImageNet-100 [TKI20] consisting of 100 classes for a total of 12.21k 256x256 images. The number of images per class varies from 1280 to 25.

Training. We use an SGD optimiser for all experiments with a weight decay of 1e-4. As for the learning rate, we utilise linear warm-up for 10 epochs that is followed by a cosine annealing schedule starting from 0.5. We train for 2000 epochs for CIFAR10-LT and CIFAR100-LT and 800 epochs for ImageNet 100-LT. For CIFAR10-LT and CIFAR100-LT we use a ResNet-18 [HZRS16] backbone. For ImageNet 100-LT we use a ResNet-50 [HZRS16] backbone. For both the MoCo [HFW+20] and the SimCLR [CKNH20] experiments, we follow [JCMW21] and use the following augmentations: resized crop, colour jitters, grey

scale and horizontal flip. MoCo details: we use a dictionary of size 10000, a projection dimensionality of 128 and a projection head with one linear layer. SimCLR details: we train with a batch size of 512 and a projection head that has two layers with an output size of 128. For evaluation, we discard the projection head and apply l2-normalisation. Regarding the proposed temperature schedules (TS), we use a period length of $T=400$ with $\tau_+=1.0$ and $\tau_-=0.1$ if not stated otherwise; for details, see Appendix F.

Evaluation. We use k nearest neighbours (kNN) and linear classifiers to assess the learned features. For kNN, we compute $l2$-normalised distances between LT samples from the train set and the class-balanced test set. For each test image, we assign it to the majority class among the top-k closest train images. We report accuracy for kNN with $k=1$ (kNN@1) and with $k=10$ (kNN@10). Compared to fine-tuning or linear probing, kNN directly evaluates the learned embedding since it relies on the learned metric and local structure of the space. We also evaluate the linear separability and generalisation of the space with a linear classifier that we train on the top of frozen backbone. For this, we consider two setups: balanced few-shot linear probing (FS LP) and long-tailed linear probing (LT LP). For FS LP, the few-shot train set is a direct subset of the original long-tailed train set with the shot number equal to the minimum class size in the original LT train set. For LT LP, we use the original LT training set; for full results, see Appendix F.

9.2.2 Effectiveness of Temperature Schedules

Contrastive Learning with TS. In Table 9.1 we present the efficacy of temperature schedules (TS) for two well-known contrastive learning frameworks MoCo [HFW+20] and SimCLR [CKNH20]. We find that both frameworks benefit from varying the temperature and we observe consistent improvements over all evaluation metrics for CIFAR10-LT and CIFAR100-LT, i.e. the local structure of the embedding space (kNN) and the global structure (linear probe) are both improved. Moreover, we show in Table 9.3 that our finding also transfers to ImageNet 100-LT. Furthermore, in Table 9.2 we evaluate the performance of the proposed method on the CIFAR10 and CIFAR100 datasets with different imbalance ratios. An imbalance ratio of 50 (imb50) reflects less pronounced imbalance, and imb150 corresponds to the datasets with only 30 (CIFAR10) and 3 (CIFAR100) samples for the smallest class. Varying τ during training improves the performance for different long-tailed data; for a discussion on the dependence of the improvement on the imbalance ratio, please see Appendix F.

	CIFAR10-LT				CIFAR100-LT			
method	kNN@1	kNN@10	FS LP	LT LP	kNN@1	kNN@10	FS LP	LT LP
MoCo	63.54	64.56	69.31	65.11	28.69	28.75	26.86	30.41
MoCo + TS	**64.99**	**65.01**	**72.87**	**66.86**	**30.31**	**29.75**	**28.97**	**32.05**
SimCLR	59.84	60.19	68.29	61.86	28.81	28.12	25.70	31.20
SimCLR + TS	**63.09**	**62.91**	**71.86**	**65.03**	**31.06**	**30.06**	**28.89**	**33.28**

Table 9.1: **Effect of temperature scheduling.** Comparison of MoCo vs MoCo+TS and SimCLR vs SimCLR+TS on CIFAR10-LT and CIFAR100-LT with kNN, few-shot and long-tail linear probe (FS LP and LT LP).

TS vs SDCLR. Further, we compare our method with SDCLR [JCMW21]. In SDCLR, SimCLR is modified s.t. the embeddings of the online model are contrasted with those of a pruned version of the same model, which is updated after every epoch. Since the pruning is done by simply masking the pruned weights of the original model, SDCLR requires twice as much memory compared to the original SimCLR and extra computational time to prune the model every epoch. In contrast, our method does not require any changes in the architecture or training. In Table 9.3 we show that this simple approach improves not only over the original SimCLR, but also over SDCLR in most metrics.

	CIFAR10-LT				CIFAR100-LT			
	imb 50		imb 150		imb 50		imb 150	
method	kNN@10	FS LP	kNN@10	FS LP	kNN@10	FS LP	kNN@10	FS LP
MoCo	69.12	74.16	59.13	65.76	32.22	33.53	25.36	22.73
MoCo + TS	**71.49**	**76.37**	**60.83**	**68.59**	**33.24**	**35.03**	**26.75**	**22.78**

Table 9.2: **Effect of imbalance ratio.** MoCo vs MoCo+TS on CIFAR10-LT and CIFAR100-LT for imbalance ratio 50 (imb50) and 150 (imb150). Evaluation metrics: kNN classifier and few-shot linear probe (FS LP).

	CIFAR10-LT			CIFAR100-LT			ImageNet-100-LT		
method	kNN@10	FS LP	LS LP	kNN@10	FS LP	LT LP	kNN@10	FS LP	LT LP
SimCLR	60.19	68.29	61.68	28.12	25.70	31.20	38.00	42.64	44.82
SDCLR	60.74	71.03	64.99	29.22	27.28	**34.23**	37.36	42.74	46.40
SimCLR + TS	**62.91**	**71.86**	**65.03**	**30.06**	**28.89**	33.28	**38.86**	**45.18**	**47.26**

Table 9.3: **Comparison with SDCLR.** SimCLR vs SDCLR vs SimCLR+TS on CIFAR10-LT, CIFAR100-LT, and ImageNet 100-LT. Evaluation: kNN classifier, few-shot (FS LP) and long-tail linear probe (LT LP).

9.2.3 Ablations

In this section, we evaluate how the hyperparameters in Equation (9.5) can influence the model behaviour.

Cosine Boundaries. First, we vary the lower τ_- and upper τ_+ bounds of τ for the cosine schedule. In Table 9.4 we assess the performance of MoCo+TS with different τ_- and τ_+ on CIFAR10 with FS LP. We observe a clear trend that with a wider range of τ values the performance increases. We attribute this to the ability of the model to learn better 'hard' features with low τ and improve semantic structure for high τ. Note that 0.07 is the value for τ in many current contrastive learning methods.

τ_- \ τ_+	0.2	0.3	0.4	0.5	1.0
0.07	69.46	68.86	71.29	71.83	**73.26**
0.1	68.17	70.34	71.25	72.31	72.87
0.2	68.89	69.37	70.12	69.65	71.42

Table 9.4: **Influence of τ_- and τ_+.** Few-shot linear probe (FS LP) results on CIFAR10 for a model trained with MoCo+TS. Performance increases with the difference between τ_- and τ_+.

TS	FS LP
fixed	68.89
step	70.18
rand	70.26
oscil	71.50
cos	**72.31**

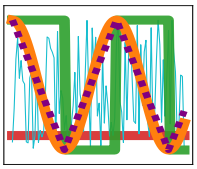


Table 9.5: **Alternative Schedules**. Constant, step function, and random sampling. All functions are bounded by 0.1 and 0.5.

Cosine Period. We investigate if the length of the period T in Equation (9.5) impacts model performance. In Table 9.6, we show that modifying the temperature τ based on the cosine schedule is beneficial during training independently of the period T. The performance varies insignificantly depending on T and consistently improves over standard fixed $\tau{=}0.2$, whereas the best performance we achieve with $T{=}400$. Even though the performance is stable with respect to the period length, it changes within one period as we show in Figure 9.4. Here, we average the accuracy of one last full period over different models trained with different T and find that the models reach the best performance around $0.7\,T$. Based on this observation, we recommend to stop training after $(n - 0.3)\,T$ epochs, with n the number of full periods.

Alternatives to Cosine Schedule. Additionally, we test different methods of varying the temperature parameter τ and report the results in Table 9.5: we examine a linearly oscillating (oscil) function, a

T	T / #epochs	FS LP
no	fixed τ	68.89
200	0.1	71.86
400	0.2	72.87
1000	0.5	72.47
2000	1.0	72.22
4000	2.0	72.10

Table 9.6: **Influence of period length T.** Improvements in the few-shot linear probe accuracy (FS LP) of MoCo+TS on CIFAR10-LT are relatively robust to T.

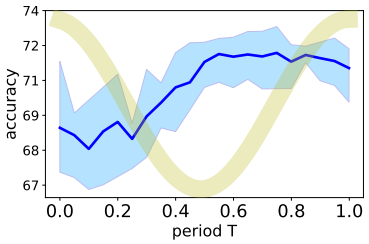


Figure 9.4: **Dependence on relative time of one period.** *Blue:* Average FS LP of last period of the models trained with $T = 200, 400, 1000, 2000$. *Light blue:* variance. *Yellow:* Relative cosine value over relative time. CIFAR10-LT trained with MoCo+TS.

step function, and random sampling. For the linear oscillations, we follow the same schedule as for the cosine version, as shown on the right of Table 9.5. For the step function, we change τ from a low (0.1) to a high (0.5) value and back every 200 epochs. For random, we uniformly sample values for τ from the range [0.1, 0.5]. In Table 9.5 we observe that both those methods for varying the τ value also improve the performance over the fixed temperature, while with the cosine schedule the model achieves the best performance. These results indicate that it is indeed the *task switching* between group-wise and instance-wise discrimination during training which is the driving factor for the observed improvements for unsupervised long-tail representation learning. We assume the reason why slow oscillation of the temperature performs better than fast (i.e. random) temperature changes is grounded in learning dynamics and the slow evolution of the embedding space during training.

9.3 Conclusion

In this work, we discover the surprising effectiveness of temperature schedules for self-supervised contrastive representation learning on imbalanced datasets. In particular, we find that a simple cosine schedule for τ consistently improves two state-of-the-art contrastive methods over several datasets and different imbalance ratios, without introducing any additional cost.

Importantly, our approach is based on a novel perspective on the contrastive loss, in which the average distance maximisation aspect is emphasised. This perspective sheds light on which samples dominate the contrastive loss and explains why large values for τ can lead to the emergence of tight clusters in the embedding space, despite the fact that individual instance *always* repel each other.

Specifically, we find that while a large τ is thus necessary to induce semantic structure, the concomitant focus on *group-wise* discrimination biases the model to encode easily separable features rather than instance-specific details. However, in long-tailed distributions, this can be particularly harmful to the most infrequent classes, as those require a higher degree of instance discrimination to remain distinguishable from the prevalent semantic categories. The proposed cosine schedule for τ overcomes this tension, by alternating between an emphasis on instance discrimination (small τ) and group-wise discrimination (large τ). As a result of this constant 'task switching', the model is trained to both structure the embedding space according to semantically meaningful features, whilst also encoding instance-specific details such that rare classes remain distinguishable from dominant ones.

10 Conclusion and Future Research Directions

Contents

If not before, at least since the advent of large scale foundation models [RKH+21, OWJ+22, BHA+21] DNN-based applications have started impacting our everyday lives. Unfortunately, however, despite the lack of DNN transparency constituting a long-standing concern especially in safety-critical situations [Rud19], recent research suggests that the latest DNN models might have become even *less interpretable* as the scale of models and data is increased [ZKB23]. While many approaches to address this 'black box' nature of DNNs have been developed [NTP+23] (cf. Chapter 2), it remains unclear whether and how reliable explanations for existing models can be obtained. It has therefore been argued that models should be explicitly designed to be more interpretable instead [Rud19, ZKB23, CLT+19, RCC+22].

In this thesis, we made a step in this direction and developed two novel inherently interpretable DNNs. By doing so, we showed that interpretability and accuracy need indeed not be at odds [RCC+22], and it is possible to design highly performant interpretable DNNs. In the following, we will summarise the key contributions of this thesis in Section 10.1, and conclude by discussing the most important insights and their relevance, as well as sketching out potential directions for future research in Section 10.2.

10.1 Summary of Key Contributions

Part I, Evaluating Attribution Methods. In Part I, we evaluated existing importance attribution methods. Specifically, in Chapter 3, we first presented a case study on explaining DNNs that were trained to predict whether a given brain scan belongs to a patient suffering from Alzheimer's disease (AD) or not. In this context, we showed that explanations derived via layer-wise relevance propagation (LRP, [BBM+16]) correlate with brain regions known to be particularly affected by AD, which can be seen as an indication that LRP can succeed in reflecting features that were relevant for the model's decision. This, however, relies on the assumption that the model indeed used these features, which we cannot say for certain.

To mitigate this issue, we devised a novel evaluation scheme for attribution methods in Chapter 4. For this, we carefully control the flow of information through the model, which allows us to identify areas that were by construction *not* used for the prediction—any attribution method that nonetheless assigns importance to those areas is thus provably not faithful. Additionally, we developed a visualisation method for qualitatively comparing attribution methods across a full dataset to gain a more holistic view of their behaviour, rather than relying only on individual input samples. Finally, we highlighted the importance of fairly comparing attribution methods. Specifically, when applying explanation methods to the same layer in a given DNN, previously reported differences between attribution methods often vanish.

Part II, Inherently Interpretable DNNs. In Part II, we developed novel DNN architectures—the CoDA and the B-cos Networks—with the explicit goal of making the decisions of these models more transparent. For this, we took inspiration from the simplicity of linear classifiers and designed the models to be

dynamic linear; i.e., so that their computations can be summarised by an equivalent linear transformation. Moreover, both of these models are optimised under carefully crafted constraints that introduce alignment pressure during training. As a result, the dynamic linear transformations of the models align with relevant signals in their inputs and these transformations are thus well-suited to be used as explanations for humans.

To achieve this, in Chapter 5 we introduced the Dynamic Alignment Units (DAUs) as a core building block for the CoDA Networks. Importantly, as the DAUs are dynamic linear themselves, models that are built exclusively from these units (i.e., the CoDA Networks) inherit this property. Moreover, the DAUs are designed to only yield large outputs if their dynamically predicted weights align with their inputs. Taking advantage of this, we restrict the overall model output and show that this encourages the dynamic linear transformation representing the CoDA Networks as a whole to also align with task-relevant input patterns.

While we found that CoDA Networks can be performant classifiers and yield highly detailed explanations for their decisions, they do not easily scale to large scale image classification problems. Therefore, in Chapter 6, we introduced the B-cos Networks, which leverage the same principles for improving model interpretability, i.e. dynamic linearity and weight-input alignment, but are significantly more efficient.

Concretely, we proposed the B-cos transformation as a replacement for the linear transformation, which is commonly used in DNNs. As a result, the B-cos transformation can easily be integrated into modern DNN architectures such as convolutional neural networks and vision transformers. Crucially, we found that the resulting B-cos Networks not only retain almost the same classification accuracy on the complex ImageNet dataset as their conventional counterparts, but also provide detailed explanations for their decisions. To the best of our knowledge, the B-cos Networks are thus the first inherently interpretable models that rival conventional models with respect to predictive accuracy on such a complex classification task.

Part III, Model Guidance. In Part III, we focused on the *utility* of model explanations. Specifically, we evaluated how explanations can be used to provide additional information to the models during training, either via annotations provided by humans (Chapter 7) or via explanations of other models (Chapter 8).

In detail, in Chapter 7 we presented an in-depth study on how to guide models via human annotations in an efficient and effective manner. Specifically, we assessed how well attribution methods localise objects in a multi-label classification setting and whether models learn to use the 'right' features in a highly biased dataset. We found that the object localisation via explanation methods can be significantly improved by regularising the model explanations via bounding box annotations on only small fraction (e.g. 1%) of the training data. Further, we showed that such regularisation can also lead to significant gains in test accuracy when the training data is highly biased. Moreover, we found that the EPG metric, which is commonly used for evaluating attribution quality, lends itself well to be used as a loss function for model guidance, as it is robust to coarse annotations and best retains a high level of detail in the resulting explanations.

Further, in Chapter 8, we extended the idea of explanation-based model guidance to the realm of knowledge distillation (KD). In particular, we proposed an explanation-enhanced KD (e^2KD) approach and showed that this can lead to significant improvements with respect to distillation fidelity: we find students trained via e^2KD to exhibit higher agreement with the teacher, to rely on the same input features for their predictions, and to provide similar explanations. I.e., the students model the function implemented by the teacher more faithfully. Moreover, we showed that this approach can even be used when only 'approximate' explanations are available, which can significantly reduce the computational cost of e^2KD.

Part IV, Self-supervised Learning on Long-Tail Data. Finally, in Part IV of this thesis, we studied how to improve contrastive representation learning techniques on imbalanced datasets. Specifically, we investigated the role of the softmax temperature τ on the learning dynamics, a hyperparameter that is typically kept fixed during training. Here, we found that τ plays a crucial role in determining the

neighbourhood of negative samples that are relevant for the loss: while a small τ only considers the local neighbourhood of a given sample, thus emphasising instance discrimination, a large τ also considers distant samples and thus also induces a global structure in the feature space.

Intuitively, less frequent classes benefit from the former, as local details are better preserved, and frequent classes from the latter, since instances with high intra-class variance can still be grouped together. Therefore, we proposed to change τ throughout the optimisation process to encourage both a good global structure, whilst also representing instance-specific details. We found that this yields significant benefits on long-tail datasets for two commonly used contrastive learning approaches at no additional cost.

While different from Parts I to III, we find that Part IV thus shares an interesting parallel with the CoDA and the B-cos Networks: in both models, we also found a *single parameter* to significantly influence the learning dynamics. In particular, both the temperature T in the CoDA Networks and the parameter B in B-cos Networks are crucial for the weight-input alignment we rely on for their interpretability. This highlights the importance of taking the optimisation process into account when interpreting DNNs, i.e., when approximating their computations via a simplified representation. Especially when done post hoc, it should be carefully assessed for every model whether a particular approximation is in fact valid.

10.2 KEY INSIGHTS AND OPEN QUESTIONS

This thesis was motivated by a desire to increase our understanding of DNNs. We approached this problem from various angles: explaining DNNs post hoc (Part I), designing them to be inherently easier to understand (Part II), investigating how to align them with our semantics and expectations (Part III), as well as studying how the loss formulation interacts with the optima the models converge to (Part II and Part IV). While the work performed for the individual chapters thus had a distinct and separate focus, we also observed some recurring themes. In the following, we highlight some of the key takeaways and connect them to interesting open questions to explore in future work.

Details Matter. While this might seem trivial, the fact that even minor details can critically influence the learning dynamics of DNNs is of particular importance for the field of explainable machine learning.

For example, throughout our work we found that even single parameters can impact what and how DNNs learn: in Chapters 5 and 6, we showed that the visual quality and the localisation ability of the model-inherent explanations is highly dependent on a single hyperparameter (the temperature T in the CoDA Networks, cf. Figure 5.5, or the parameter B in B-cos Networks, cf. Figure 6.4). Moreover, we showed that depending on how the model is trained, it could be encouraged to use relatively constant input features to compute implicit biases ('You get what you optimise for', cf. Figure 6.14). Similarly, in Chapter 9, we found that whether instance-specific details of infrequent classes are represented in the learnt embeddings depends on the temperature parameter of the loss in contrastive representation learning.

Crucially, these findings thus challenge post-hoc explanation approaches on a fundamental level. In particular, they highlight the importance of taking the specifics of the optimisation procedure into account for understanding the decisions (Chapters 5 and 6) as well as the representations (Chapter 9) of DNNs. Being agnostic to the optimisation, however, is the defining feature of post-hoc explanation approaches. That said, given their practical relevance and the fact that post-hoc explanations *have* been found to correlate with meaningful patterns in the input data (cf. [SML+21], see also our case study in Chapter 3), we believe it to be an important future research direction to understand under which circumstances post-hoc explanations could still yield reliable and useful insights into the decision process of a DNN.

In this context, it is further important to understand what exactly we can learn from a given explanation

method and why one might seem to work better than others in a specific setting. For example, in our experiments in Chapters 5 and 6 we optimise the models with the binary cross-entropy loss rather than with the more commonly used cross-entropy loss, to obtain class-specific explanations that are independent of the class logits for other classes. In recent preliminary experiments, we find that the loss function (i.e., BCE vs. CE) is indeed an important component for obtaining easily understandable explanations, and might affect model-inherent and post-hoc explanations alike. Better understanding the interplay between the loss function and the quality of model explanations constitutes an important direction for future work.

Performant and Inherently Interpretable Models are Possible. We showed that it is possible to design DNNs that are inherently interpretable without sacrificing classification accuracy. Specifically, the B-cos Networks essentially perform on par with conventional models, whilst computing their output in a much more transparent manner. Importantly, they do so without increasing the number of parameters with respect to conventional models, they can be optimised with standard training recipes, and they are compatible with state-of-the-art architectures. As such, B-cos Networks constitute a strong alternative to conventional DNNs, especially in applications in which model interpretability is of particular concern.

That said, various open questions and challenges remain. For example, this thesis focused on interpretability in the context of image classifiers. This, of course, only covers a fraction of DNN-based applications. Extending the benefits of CoDA and B-cos Networks to other tasks (e.g., object detection, image segmentation), pretraining paradigms (e.g., self-supervised learning), and modalities (e.g., audio, text) constitutes an important next step for establishing inherently interpretable DNNs firmly in the machine learning landscape. Moreover, in Chapters 5 and 6, we argue that the dynamic linear transformations obtained by CoDA and B-cos Networks are well suited to be used as explanations for humans as they resemble relevant aspects of the input signal and localise those well. While this might seem intuitive, we have not explicitly evaluated their utility in studies with human participants such as in [KMR+22]. In future work, it should be investigated how to best leverage the dynamic linear transformation in order to present easily understandable and highly informative explanations to humans for specific use cases.

Moreover, while the added cost for switching from conventional to B-cos models might be relatively low when training *from scratch*, in recent years it has become common practice to rely on pretrained, so-called 'foundational' models [RKH+21, OWJ+22, BHA+21]. Training such models can cost millions of dollars [LLH+24], and can be a major hurdle for a more wide-spread adoption of any novel type of DNN models, and B-cos Networks are certainly no exception to this. Therefore, we believe an important next challenge to solve is not only how to integrate inherent interpretability into existing network *architectures* as we did in Chapter 6, but also leverage their pretrained *weights*. Especially in the case of the B-cos Networks, we believe this to be a promising route, as these models primarily differ from conventional models in the exponent of the cosine term (see Chapter 6). It might thus be possible to 'B-cosify' conventional models by finetuning them with an increased exponent B and thereby significantly improve their interpretability.

An alternative for increasing the interpretability of foundational models is to build hybrid models as we did for the CoDA Networks in Chapter 5, i.e., to add light-weight interpretable classifiers on top of the powerful representations learnt by conventional models. This is particularly interesting in the context of the emergent field of concept-extraction methods (cf. Chapter 2). Specifically, processing the extracted concepts by an interpretable classifier might yield better estimates of the importance of a particular concept for the models' decisions, thus allowing us to obtain a more global understanding of the strategies a DNN employs to solve a given task; see also [ADE+23] for a recent work on concept-based explanations.

Beyond CoDA and B-cos Networks. We presented two approaches that leverage dynamic linearity and alignment pressure to increase the interpretability of DNNs. While the CoDA Networks constituted a crucial proof of concept, the B-cos Networks distilled the most relevant aspects of CoDA Networks to lift

the performance of dynamic-linearity-based interpretable models to a level that rivals conventional DNNs. That said, the B-cos Networks are still only the second iteration of such models. In future work, we plan to study how to leverage insights gained from the B-cos Networks to further improve the performance and the interpretability of our proposed models.

For example, as we discuss in Appendix C, the B-cos Networks exhibit interesting parallels to radial basis function (RBFs) Networks. While DNNs have proven more successful on complex tasks, single-layer RBF Networks have been shown to exhibit other desirable properties: e.g., they are more robust to adversarial examples [GSS15] and inherently provide low confidence predictions far from the training data [GSS15, HAB19]. Interestingly, we find that B-cos Networks can be understood as a multi-layer variant of RBF Networks. While designing multi-layer RBF models has been attempted before [HW19, ZHS18], to the best of our knowledge, the B-cos Networks are the first version of deep RBF-like models to yield competitive performance on complex datasets such as ImageNet. In future work, we plan to further analyse this relation and the implications that it carries.

In summary, this thesis contributes towards increasing the interpretability of Deep Learning approaches, both by better understanding existing models and explanation methods, but, crucially, also by showing that making DNNs interpretable *by design* is possible at relatively minor costs in performance.

We are convinced that this constitutes an important step towards making the decision processes of DNNs more transparent, and that it opens up an exciting array of research directions to explore in the future.

BIBLIOGRAPHY

[AAES+23] Sajid Ali, Tamer Abuhmed, Shaker El-Sappagh, Khan Muhammad, Jose M Alonso-Moral, Roberto Confalonieri, Riccardo Guidotti, Javier Del Ser, Natalia Díaz-Rodríguez, and Francisco Herrera. Explainable Artificial Intelligence (XAI): What we know and what is left to attain Trustworthy Artificial Intelligence. *Information Fusion*, 99, 2023. Cited on page 9.

[ADD+22] Mohamed Afham, Isuru Dissanayake, Dinithi Dissanayake, Amaya Dharmasiri, Kanchana Thilakarathna, and Ranga Rodrigo. Crosspoint: Self-supervised cross-modal contrastive learning for 3d point cloud understanding. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. Cited on page 18.

[ADE+23] Reduan Achtibat, Maximilian Dreyer, Ilona Eisenbraun, Sebastian Bosse, Thomas Wiegand, Wojciech Samek, and Sebastian Lapuschkin. From attribution maps to human-understandable explanations through Concept Relevance Propagation. *Nature Machine Intelligence*, 5(9), 2023. Cited on pages 11, 12, and 142.

[ADPGG21] Anna Arias-Duart, Ferran Parés, and Dario Garcia-Gasulla. Focus! Rating XAI Methods and Finding Biases with Mosaics. *arXiv:2109.15035*, 2021. Cited on pages 13, 44, 45, 51, and 52.

[AGM+18] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity Checks for Saliency Maps. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. Cited on pages 12, 13, 15, 37, 59, 67, 68, 194, 198, 199, and 205.

[AHMJ+14] Ossama Abdel-Hamid, Abdel-Rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, and Dong Yu. Convolutional neural networks for speech recognition. *IEEE/ACM Transactions on audio, speech, and language processing*, 22(10), 2014. Cited on page 25.

[AHY+20] Md Zahangir Alom, Mahmudul Hasan, Chris Yakopcic, Tarek M. Taha, and Vijayan K. Asari. Improved inception-residual convolutional neural network for object recognition. *Neural Computing and Applications*, 2020. Cited on page 66.

[AJ18] David Alvarez-Melis and Tommi S. Jaakkola. Towards Robust Interpretability with Self-Explaining Neural Networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. Cited on pages 4, 13, 15, and 66.

[AKK+22] Saeid Asgari, Aliasghar Khani, Fereshte Khani, Ali Gholami, Linh Tran, Ali Mahdavi-Amiri, and Ghassan Hamarneh. MaskTune: Mitigating Spurious Correlations by Forcing to Explore. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. Cited on page 16.

[AKW+23] Léo Andéol, Yusei Kawakami, Yuichiro Wada, Takafumi Kanamori, Klaus-Robert Müller, and Grégoire Montavon. Learning domain invariant representations by joint Wasserstein distance minimization. *Neural Networks*, 2023. Cited on page 190.

[AMSA21] Stephan Alaniz, Diego Marcos, Bernt Schiele, and Zeynep Akata. Learning decision trees recurrently through communication. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. Cited on page 14.

[ANS+21] Christopher J. Anders, David Neumann, Wojciech Samek, Klaus-Robert Müller, and Sebastian Lapuschkin. Software for Dataset-wide XAI: From Local Explanations to Global Insights with Zennit, CoRelAy, and ViRelAy. *arXiv:2106.13200*, 2021. Cited on page 190.

[ARV19] Yuki Markus Asano, Christian Rupprecht, and Andrea Vedaldi. Self-labelling via simultaneous clustering and representation learning. In *International Conference on Learning Representations (ICLR)*, 2019. Cited on pages 18 and 129.

[AVT21] R. Alharbi, M. N. Vu, and M. T. Thai. Learning Interpretation with Explainable Knowledge Distillation. In *2021 IEEE International Conference on Big Data (Big Data)*, Los Alamitos, CA, USA, 2021. IEEE Computer Society. Cited on page 17.

[AZ20] Samira Abnar and Willem Zuidema. Quantifying Attention Flow in Transformers. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020. Cited on page 91.

[Bac80] Jürgen Backhaus. The pareto principle. *Analyse & Kritik*, 2(2), 1980. Cited on page 105.

[BB91] H Braak and E Braak. Neuropathological stageing of Alzheimer-related changes. *Acta Neuropathologica*, 82(4), 1991. Cited on page 37.

[BB19] Wieland Brendel and Matthias Bethge. Approximating CNNs with Bag-of-local-Features models works surprisingly well on ImageNet. In *International Conference on Learning Representations (ICLR)*, 2019. Cited on page 14.

[BBH+21] Alexander Binder, Michael Bockmayr, Miriam Hägele, Stephan Wienert, Daniel Heim, Katharina Hellweg, Masaru Ishii, Albrecht Stenzinger, Andreas Hocke, Carsten Denkert, et al. Morphological and molecular breast cancer profiling through explainable machine learning. *Nature Machine Intelligence*, 3(4), 2021. Cited on page 3.

[BBM+15] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7), 2015. Cited on pages 3, 4, 10, 11, 24, 26, 45, 52, and 190.

[BBM+16] Alexander Binder, Sebastian Bach, Gregoire Montavon, Klaus-Robert Müller, and Wojciech Samek. Layer-Wise Relevance Propagation for Deep Neural Network Architectures. In *International Conference on Information Science and Applications (ICISA)*. Springer Singapore, 2016. Cited on pages 26, 28, and 139.

[BDSS21a] Andy Brock, Soham De, Samuel L Smith, and Karen Simonyan. High-Performance Large-Scale Image Recognition Without Normalization. In *International Conference on Machine Learning (ICML)*, 2021. Cited on page 214.

[BDSS21b] Andy Brock, Soham De, Samuel L Smith, and Karen Simonyan. High-Performance Large-Scale Image Recognition Without Normalization. In *International Conference on Machine Learning (ICML)*, volume 139 of *Proceedings of Machine Learning Research*. PMLR, 2021. Cited on page 258.

[BES17] Mark W Bondi, Emily C Edmonds, and David P Salmon. Alzheimer's disease: Past, present, and future. *Journal of the International Neuropsychological Society*, 23(9-10), 2017. Cited on page 23.

[BEWR19] Moritz Böhle, Fabian Eitel, Martin Weygandt, and Kerstin Ritter. Layer-wise Relevance Propagation for Explaining Deep Neural Network Decisions in MRI-based Alzheimer's Disease Classification. *Frontiers in Aging Neuroscience*, 11, 2019. Cited on pages 4, 5, and 24.

[BF20] Jasmijn Bastings and Katja Filippova. The elephant in the interpretability room: Why use attention as explanation when we have saliency methods? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, 2020. Cited on page 15.

[BFS21] Moritz Böhle, Mario Fritz, and Bernt Schiele. Convolutional Dynamic Alignment Networks for Interpretable Classifications. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. Cited on pages 5, 6, 12, 41, 44, 58, 86, 172, 173, 179, 186, 187, 190, 191, 212, 215, and 217.

[BFS22a] Moritz Böhle, Mario Fritz, and Bernt Schiele. Optimising for Interpretability: Convolutional Dynamic Alignment Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2022. Cited on pages 5, 6, and 58.

[BFS22b] Moritz Böhle, Mario Fritz, and Bernt Schiele. B-cos Networks: Alignment is All we Need for Interpretability. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. Cited on pages 6, 16, 78, 100, 102, 103, 105, 115, 120, 237, 250, and 258.

[BHA+21] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, S. Buch, Dallas Card, Rodrigo Castellon, Niladri S. Chatterji, Annie S. Chen, Kathleen A. Creel, Jared Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren E. Gillespie, Karan Goel, Noah D. Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas F. Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, O. Khattab, Pang Wei Koh, Mark S. Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir P. Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Benjamin Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, J. F. Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Robert Reich, Hongyu Ren, Frieda Rong, Yusuf H. Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishna Parasuram Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei A. Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. On the Opportunities and Risks of Foundation Models. *arXiv:2108.07258*, 2021. Cited on pages 139 and 142.

[BJV+19] Maxim Berman, Hervé Jégou, Andrea Vedaldi, Iasonas Kokkinos, and Matthijs Douze. MultiGrain: a unified image embedding for classes and instances. *arXiv:1902.05509*, 2019. Cited on page 215.

[BKH16] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv:1607.06450*, 2016. Cited on page 83.

[BL88] David S Broomhead and David Lowe. *Radial Basis Functions, Multi-Variable Functional Interpolation and Adaptive Networks*, volume 4148 of *RSRE memorandum*. Royals Signals & Radar Establishment Worcestershire, 1988. Cited on page 215.

[BML+16] Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, Klaus-Robert Müller, and Wojciech Samek. Layer-wise relevance propagation for neural networks with local renormalization layers. In *International Conference on Artificial Neural Networks (ICANN)*. Springer, 2016. Cited on page 26.

[BNK+19] Gagan Bansal, Besmira Nushi, Ece Kamar, Daniel S Weld, Walter S Lasecki, and Eric Horvitz. Updates in Human-AI Teams: Understanding and Addressing the Performance/Compatibility Tradeoff. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 33, 2019. Cited on page 115.

[BPBPQE23] Jenny Benois-Pineau, Romain Bourqui, Dragutin Petkovic, and Georges Quenot (Eds.). *Explainable Deep Learning AI: Methods and Challenges*. Elsevier, 2023. Cited on page 9.

[BPL22] Adrien Bardes, Jean Ponce, and Yann LeCun. VICReg: Variance-Invariance-Covariance Regularization For Self-Supervised Learning. In *International Conference on Learning Representations (ICLR)*, 2022. Cited on page 18.

[BSFS24] Moritz Böhle, Navdeeppal Singh, Mario Fritz, and Bernt Schiele. B-cos Alignment for Inherently Interpretable CNNs and Vision Transformers. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2024. Cited on pages 6, 16, 78, 105, 117, 118, 236, 237, 250, 258, 259, and 260.

[BSH+10] David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. How to explain individual classification decisions. *The Journal of Machine Learning Research (JMLR)*, 2010. Cited on pages 67, 89, 205, and 215.

[BTK15] Rembrandt Bakker, Paul Tiesinga, and Rolf Kötter. The scalable brain atlas: instant web-based access to public brain atlases and related content. *Neuroinformatics*, 13(3), 2015. Cited on pages 24, 25, 27, and 36.

[BZK+17] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. Cited on pages 11 and 19.

[BZK22] Lucas Beyer, Xiaohua Zhai, and Alexander Kolesnikov. Better plain ViT baselines for ImageNet-1k. *arXiv:2205.01580*, 2022. Cited on pages 86, 90, 92, and 215.

[BZR+22] Lucas Beyer, Xiaohua Zhai, Amélie Royer, Larisa Markeeva, Rohan Anil, and Alexander Kolesnikov. Knowledge Distillation: A Good Teacher Is Patient and Consistent. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. Cited on pages 17, 115, 119, 120, 121, 122, 255, and 258.

[Cal23a] California Public Utilities Commission (CPUC). Resolution TL-19144. https://docs.cpuc.ca.gov/PublishedDocs/Published/G000/M516/K812/516812344.PDF, 2023. Accessed: 2023-09-06. Cited on page 1.

[Cal23b] California Public Utilities Commission (CPUC). Resolution TL-19145. https://docs.cpuc.ca.gov/PublishedDocs/Published/G000/M516/K812/516812218.PDF, 2023. Accessed: 2023-09-06. Cited on page 1.

[Cas16] Davide Castelvecchi. Can we open the black box of AI? *Nature*, 538(7623), 2016. Cited on page 23.

[CBR20] Zhi Chen, Yijie Bei, and Cynthia Rudin. Concept whitening for interpretable image recognition. *Nature Machine Intelligence*, 2(12), 2020. Cited on page 11.

[CFCS22] Julien Colin, Thomas Fel, Rémi Cadène, and Thomas Serre. What I Cannot Predict, I Do Not Understand: A Human-Centered Evaluation Framework for Explainability Methods. *Advances in Neural Information Processing Systems (NeurIPS)*, 35, 2022. Cited on page 12.

[CFGH20] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved Baselines with Momentum Contrastive Learning. *arXiv:2003.04297*, 2020. Cited on pages 18 and 129.

[CGW21] Hila Chefer, Shir Gur, and Lior Wolf. Transformer Interpretability Beyond Attention Visualization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. Cited on page 15.

[CH21] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. Cited on page 18.

[CJL+19] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. Cited on page 135.

[CKNH20] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning (ICML)*, 2020. Cited on pages 18, 129, 131, 135, 136, and 265.

[CKS+20] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. Big self-supervised models are strong semi-supervised learners. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. Cited on page 18.

[CLL21] Ting Chen, Calvin Luo, and Lala Li. Intriguing properties of contrastive losses. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. Cited on pages 19 and 265.

[CLT+19] Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. This Looks Like That: Deep Learning for Interpretable Image Recognition. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. Cited on pages 14 and 139.

[CLY+15] Chunshui Cao, Xianming Liu, Yi Yang, Yinan Yu, Jiang Wang, Zilei Wang, Yongzhen Huang, Liang Wang, Chang Huang, Wei Xu, et al. Look and Think Twice: Capturing Top-Down Visual Attention with Feedback Convolutional Neural Networks. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015. Cited on pages 12 and 13.

[CLZJ21] Pengguang Chen, Shu Liu, Hengshuang Zhao, and Jiaya Jia. Distilling Knowledge via Knowledge Review. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. Cited on pages 17, 120, 121, and 255.

[CMM+20] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. Cited on pages 18, 129, and 265.

[Coa17] Patrick Coady. VGGNet and Tiny ImageNet. https://learningai.io/projects/2017/06/29/tiny-imagenet.html, 2017. Accessed: 2020-11-08. Cited on page 66.

[Cou21] Council of the European Union. Proposal for a Regulation of the European Parliament and of the Council Laying Down Harmonised Rules on Artificial Intelligence (Artificial Intelligence Act) and Amending Certain Union Legistlative Acts. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A52021PC0206&qid=1694009083609, 2021. Accessed: 2023-09-06. Cited on pages 1 and 115.

[CSHB18] Aditya Chattopadhyay, Anirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian. Grad-CAM++: Improved Visual Explanations for Deep Convolutional Networks. In *Winter Conference on Applications of Computer Vision (WACV)*, 2018. Cited on pages 10, 12, and 45.

[CSW22] Hila Chefer, Idan Schwartz, and Lior Wolf. Optimizing Relevance Maps of Vision Transformers Improves Robustness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. Cited on page 16.

[CTM+21] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. Cited on pages 3 and 118.

[CWW+11] Ramon Casanova, Christopher T Whitlow, Benjamin Wagner, Jeff Williamson, Sally A Shumaker, Joseph A Maldjian, and Mark A Espeland. High Dimensional Classification of Structural MRI Alzheimer's Disease Data Based on Large Scale Regularization. *Frontiers in Neuroinformatics*, 5, 2011. Cited on pages 36 and 37.

[CZSL20a] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. Cited on pages 203 and 204.

[CZSL20b] Ekin Dogus Cubuk, Barret Zoph, Jon Shlens, and Quoc Le. RandAugment: Practical Automated Data Augmentation with a Reduced Search Space. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33. Curran Associates, Inc., 2020. Cited on page 259.

[DAA+19] Ann-Kathrin Dombrowski, Maximillian Alber, Christopher Anders, Marcel Ackermann, Klaus-Robert Müller, and Pan Kessel. Explanations can be manipulated and geometry is to blame. *Advances in Neural Information Processing Systems (NeurIPS)*, 32, 2019. Cited on page 13.

[DBC22] Jon Donnelly, Alina Jade Barnett, and Chaofan Chen. Deformable ProtoPNet: An Interpretable Image Classifier Using Deformable Prototypes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. Cited on page 14.

[DBK+21] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations (ICLR)*, 2021. Cited on pages 15, 53, 78, 86, and 105.

[DCH+09] R. S. Desikan, H. J. Cabral, C. P. Hess, W. P. Dillon, C. M. Glastonbury, M. W. Weiner, N. J. Schmansky, D. N. Greve, D. H. Salat, R. L. Buckner, and B. Fischl. Automated MRI measures identify individuals with mild cognitive impairment and Alzheimer's disease. *Brain*, 132(8), 2009. Cited on page 36.

[DDS+09] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. Cited on pages 44, 64, 77, 85, 105, 116, 118, 129, 191, 236, 237, and 258.

[DG17] Piotr Dabkowski and Yarin Gal. Real Time Image Saliency for Black Box Classifiers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. Cited on page 10.

[DR20] Saurabh Desai and Harish G. Ramaswamy. Ablation-CAM: Visual Explanations for Deep Convolutional Network via Gradient-free Localization. In *Winter Conference on Applications of Computer Vision (WACV)*, 2020. Cited on pages 10 and 45.

[DSA+01] An-Tao Du, Norbert Schuff, Diane Amend, Mikko Laakso, Yuan-Yu Hsu, William Jagust, Kristine Yaffe, Joel Kramer, Bruce Reed, David Norman, Helena Chui, and Michael Weiner. Magnetic resonance imaging of the entorhinal cortex and hippocampus in mild cognitive impairment and Alzheimer's disease. *Journal of Neurology, Neurosurgery & Psychiatry*, 71(4), 2001. Cited on page 36.

[DSK+19] Yiming Ding, Jae Ho Sohn, Michael G. Kawczynski, Hari Trivedi, Roy Harnish, Nathaniel W. Jenkins, Dmytro Lituiev, Timothy P. Copeland, Mariam S. Aboian, Carina Mari Aparici, Spencer C. Behr, Robert R. Flavell, Shih-Ying Huang, Kelly A. Zalocusky, Lorenzo Nardo, Youngho Seo, Randall A. Hawkins, Miguel Hernandez Pampaloni, Dexter Hadley, and Benjamin L. Franc. A Deep Learning Model to Predict a Diagnosis of Alzheimer Disease by Using 18 F-FDG PET of the Brain. *Radiology*, 290(2), 2019. Cited on page 24.

[DT17] Terrance Devries and Graham W. Taylor. Improved Regularization of Convolutional Neural Networks with Cutout. *arXiv:1708.04552*, 2017. Cited on page 215.

[DYC+20] Yu Deng, Jiaolong Yang, Dong Chen, Fang Wen, and Xin Tong. Disentangled and controllable face image generation via 3d imitative-contrastive learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. Cited on page 18.

[EBCV09] Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3), 2009. Cited on page 11.

[EBPA18] Soheil Esmaeilzadeh, Dimitrios Ioannis Belivanis, Kilian M Pohl, and Ehsan Adeli. End-To-End Alzheimer's Disease Diagnosis and Biomarker Identification. In *International Workshop on Machine Learning in Medical Imaging*. Springer, 2018. Cited on page 24.

[EJS+21] Gabriel Erion, Joseph D Janizek, Pascal Sturmfels, Scott M Lundberg, and Su-In Lee. Improving Performance of Deep Learning Models with Axiomatic Attribution Priors and Expected Gradients. *Nature Machine Intelligence*, 3(7), 2021. Cited on page 16.

[ESSS21] Aleksandr Ermolov, Aliaksandr Siarohin, Enver Sangineto, and Nicu Sebe. Whitening for self-supervised representation learning. In *International Conference on Machine Learning (ICML)*, 2021. Cited on page 18.

[EVGW+09] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision (IJCV)*, 88, 2009. Cited on pages 100, 105, 116, 123, and 236.

[EVGW+12] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html, 2012. Cited on page 259.

[FBB+23] Thomas FEL, Victor Boutin, Louis Béthune, Remi Cadene, Mazda Moayeri, Léo Andéol, Mathieu Chalvidal, and Thomas Serre. A Holistic Approach to Unifying Automatic Concept Extraction and Concept Importance Estimation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. Cited on pages 11 and 12.

[FFJJ+10] Giovanni B Frisoni, Nick C Fox, Clifford R Jack Jr, Philip Scheltens, and Paul M Thompson. The clinical use of structural MRI in Alzheimer disease. *Nature Reviews Neurology*, 6(2), 2010. Cited on pages 23 and 36.

[FHYF19] Hiroshi Fukui, Tsubasa Hirakawa, Takayoshi Yamashita, and Hironobu Fujiyoshi. Attention Branch Network: Learning of Attention Mechanism for Visual Explanation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. Cited on page 16.

[FPB+23] Thomas Fel, Agustin Picard, Louis Bethune, Thibaut Boissin, David Vigouroux, Julien Colin, Rémi Cadène, and Thomas Serre. CRAFT: Concept Recursive Activation Factorization for Explainability. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. Cited on pages 11 and 12.

[FPM+23] Fartash Faghri, Hadi Pouransari, Sachin Mehta, Mehrdad Farajtabar, Ali Farhadi, Mohammad Rastegari, and Oncel Tuzel. Reinforce Data, Multiply Impact: Improved Model Accuracy and Robustness with Dataset Reinforcement. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. Cited on pages 18 and 119.

[FRRLS22] Thomas Fel, Ivan F Rodriguez Rodriguez, Drew Linsley, and Thomas Serre. Harmonizing the Object Recognition Strategies of Deep Neural Networks with Humans. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. Cited on page 16.

[FSSK22] Felix Friedrich, Wolfgang Stammer, Patrick Schramowski, and Kristian Kersting. A Typology to Explore and Guide Explanatory Interactive Machine Learning. *arXiv:2203.03668*, 2022. Cited on page 16.

[FT19] William Falcon and The PyTorch Lightning team. PyTorch Lightning, 2019. Cited on page 258.

[FTP+22] Patrick Fernandes, Marcos Treviso, Danish Pruthi, André FT Martins, and Graham Neubig. Learning to Scaffold: Optimizing Model Explanations for Teaching. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. Cited on page 16.

[FV17] Ruth C Fong and Andrea Vedaldi. Interpretable Explanations of Black Boxes by Meaningful Perturbation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017. Cited on pages 10, 12, 13, and 103.

[FV18] Ruth Fong and Andrea Vedaldi. Net2vec: Quantifying and explaining how concepts are encoded by filters in deep neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. Cited on page 19.

[FVHC+17] Daniel Ferreira, Chloë Verhagen, Juan Andrés Hernández-Cabrera, Lena Cavallin, Chun-Jie Guo, Urban Ekman, J-Sebastian Muehlboeck, Andrew Simmons, José Barroso, Lars-Olof Wahlund, et al. Distinct subtypes of Alzheimer's disease based on patterns of brain atrophy: longitudinal trajectories and clinical applications. *Scientific reports*, 7, 2017. Cited on pages 30 and 37.

[FZ20] Vitaly Feldman and Chiyuan Zhang. What Neural Networks Memorize and Why: Discovering the Long Tail via Influence Estimation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. Cited on page 99.

[Gab46] Dennis Gabor. Theory of communication. Part 1: The analysis of information. *Journal of the Institution of Electrical Engineers-part III: radio and communication engineering*, 93(26), 1946. Cited on page 11.

[GAM13] Ashish Gupta, Murat Ayhan, and Anthony Maida. Natural image bases to represent neuroimaging data. In *International Conference on Machine Learning (ICML)*. PMLR, 2013. Cited on pages 23 and 25.

[GBP+21] Spyros Gidaris, Andrei Bursuc, Gilles Puy, Nikos Komodakis, Matthieu Cord, and Patrick Perez. OBoW: Online Bag-of-Visual-Words Generation for Self-Supervised Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. Cited on page 18.

[GBY+18] Leilani H Gilpin, David Bau, Ben Z Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*. IEEE, 2018. Cited on page 2.

[GCV+21] Gabriel Goh, Nick Cammarata, Chelsea Voss, Shan Carter, Michael Petrov, Ludwig Schubert, Alec Radford, and Chris Olah. Multimodal neurons in artificial neural networks. *Distill*, 6(3), 2021. Cited on page 11.

[GNO+23] Mara Graziani, An-phi Nguyen, Laura O'Mahony, Henning Müller, and Vincent Andrearczyk. Concept discovery and dataset exploration with singular value decomposition. In *International Conference on Learning Representations (ICLR), Workshop*, 2023. Cited on page 11.

[Gra78] Goesta H Granlund. In search of a general picture processing operator. *Computer Graphics and Image Processing*, 8(2), 1978. Cited on page 11.

[GS22] Matthew Gwilliam and Abhinav Shrivastava. Beyond Supervised vs. Unsupervised: Representative Benchmarking and Analysis of Image Representation Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. Cited on page 18.

[GSA+20] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. Cited on page 18.

[GSB+22] Yuyang Gao, Tong Steven Sun, Guangji Bai, Siyi Gu, Sungsoo Ray Hong, and Zhao Liang. RES: A Robust Framework for Guiding Visual Explanation. In *Conference on Knowledge Discovery and Data Mining (KDD)*, 2022. Cited on pages 16, 17, 100, 102, 103, 104, and 105.

[GSS15] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. In *International Conference on Learning Representations*, 2015. Cited on pages 16, 143, and 216.

[GSZH22] Yuyang Gao, Tong Steven Sun, Liang Zhao, and Sungsoo Ray Hong. Aligning Eyes between Humans and Deep Neural Network through Interactive Attention Alignment. *Proceedings of the ACM on Human-Computer Interaction*, 6(CSCW2), 2022. Cited on pages 16, 100, 102, 104, and 105.

[GWFM+13] Ian Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. Maxout networks. In *International Conference on Machine Learning (ICML)*, 2013. Cited on page 83.

[GWZK19] Amirata Ghorbani, James Wexler, James Y Zou, and Been Kim. Towards automatic concept-based explanations. *Advances in Neural Information Processing Systems (NeurIPS)*, 32, 2019. Cited on page 11.

[GYLL23] Ziyao Guo, Haonan Yan, Hui Li, and Xiaodong Lin. Class Attention Transfer Based Knowledge Distillation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. Cited on pages 17, 120, 121, and 255.

[GZB+24] Robert Geirhos, Roland S Zimmermann, Blair Bilodeau, Wieland Brendel, and Been Kim. Don't trust your eyes: on the (un)reliability of feature visualizations. In *International Conference on Learning Representations (ICLR)*, 2024. Cited on page 11.

[GZF+19] Hao Guo, Kang Zheng, Xiaochuan Fan, Hongkai Yu, and Song Wang. Visual Attention Consistency under Image Transforms for Multi-Label Image Classification. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. Cited on page 16.

[HAB19] Matthias Hein, Maksym Andriushchenko, and Julian Bitterwolf. Why ReLU networks yield high-confidence predictions far away from the training data and how to mitigate the problem. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. Cited on pages 143 and 216.

[HB20] Peter Hase and Mohit Bansal. Evaluating explainable AI: Which algorithmic explanations help users predict model behavior? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online, 2020. Association for Computational Linguistics. Cited on page 12.

[HBH+20] Elad Hoffer, Tal Ben-Nun, Itay Hubara, Niv Giladi, Torsten Hoefler, and Daniel Soudry. Augment Your Batch: Improving Generalization Through Instance Repetition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. Cited on page 215.

[HCLR19] Peter Hase, Chaofan Chen, Oscar Li, and Cynthia Rudin. Interpretable image recognition with hierarchical prototypes. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 7, 2019. Cited on page 14.

[HCMN22] Misgina Tsighe Hagos, Kathleen M Curran, and Brian Mac Namee. Identifying Spurious Correlations and Correcting them with an Explanation-based Learning. *arXiv:2211.08285*, 2022. Cited on page 16.

[HCX+22] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. Cited on page 18.

[HEKK19] Sara Hooker, Dumitru Erhan, Pieter-Jan Kindermans, and Been Kim. A benchmark for interpretability methods in deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. Cited on page 13.

[HFRK21] Adrian Hoffmann, Claudio Fanconi, Rahul Rade, and Jonas Kohler. This Looks Like That... Does it? Shortcomings of Latent Space Prototype Interpretability in Deep Networks, 2021. Cited on page 14.

[HFW+20] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum Contrast for Unsupervised Visual Representation Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. Cited on pages 18, 131, 135, and 136.

[HLVDMW17] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely Connected Convolutional Networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. Cited on pages 59, 66, 78, 86, 105, and 120.

[HMRGP14] Antonio R Hidalgo-Muñoz, Javier Ramírez, Juan M Górriz, and Pablo Padilla. Regions of interest computed by SVM wrapped method for Alzheimer's disease examination from segmented MRI. *Frontiers in Aging Neuroscience*, 6, 2014. Cited on page 36.

[HOWT06] Geoffrey Hinton, Simon Osindero, Max Welling, and Yee-Whye Teh. Unsupervised discovery of nonlinear structure using contrastive backpropagation. *Cognitive science*, 30(4), 2006. Cited on page 11.

[HR19] Lars Kai Hansen and Laura Rieger. Interpretability in intelligent systems–a new concept? *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, 2019. Cited on page 2.

[HSMR21] Robin Hesse, Simone Schaub-Meyer, and Stefan Roth. Fast Axiomatic Attribution for Neural Networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. Cited on pages 103, 105, and 236.

[HSMR23] Robin Hesse, Simone Schaub-Meyer, and Stefan Roth. FunnyBirds: A synthetic vision dataset for a part-based analysis of explainable AI methods. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. Cited on pages 12 and 14.

[HVD15] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the Knowledge in a Neural Network. In *Advances in Neural Information Processing Systems (NeurIPS), Workshop*, 2015. Cited on pages 17, 115, 121, 123, and 255.

[HW19] Andrew Hryniowski and Alexander Wong. DeepLABNet: End-to-end Learning of Deep Radial Basis Networks with Fully Learnable Basis Functions. *arXiv:1911.09257*, 2019. Cited on pages 16, 143, 215, and 216.

[HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. Cited on pages 44, 47, 59, 66, 67, 72, 73, 78, 86, 87, 88, 105, 120, 135, 172, 174, 180, 182, 184, 187, 204, 214, 229, and 260.

[Ide18] Yerlan Idelbayev. "Proper ResNet Implementation for CIFAR10/CIFAR100 in PyTorch". https://github.com/akamaster/pytorch_resnet_cifar10, 2018. Accessed: 2020-06-05. Cited on pages 73, 203, and 204.

[IS15] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *International Conference on Machine Learning (ICML)*, 2015. Cited on pages 49, 69, 73, 83, 187, and 203.

[ITAC18] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondřej Chum. Mining on manifolds: Metric learning without labels. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. Cited on page 18.

[JCMW21] Ziyu Jiang, Tianlong Chen, Bobak Mortazavi, and Zhangyang Wang. Self-Damaging Contrastive Learning. In *International Conference on Machine Learning (ICML)*, 2021. Cited on pages 19, 135, 136, and 262.

[JDBTG16] Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc V Gool. Dynamic filter networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016. Cited on pages 15 and 62.

[JEP+21] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873), 2021. Cited on page 1.

[JGB+21] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *International Conference on Machine Learning (ICML)*. PMLR, 2021. Cited on page 1.

[JH20] Bohan Jia and Qiyu Huang. DE-CapsNet: A Diverse Enhanced Capsule Network with Disperse Dynamic Routing. *Applied Sciences*, 2020. Cited on page 66.

[Joh16] Justin Johnson. Tiny ImageNet Visual Recognition Challenge. https://github.com/jcjohnson/tiny-imagenet, 2016. Accessed: 2020-11-10. Cited on page 64.

[JZH+21] Peng-Tao Jiang, Chang-Bin Zhang, Qibin Hou, Ming-Ming Cheng, and Yunchao Wei. LayerCAM: Exploring Hierarchical Class Activation Maps for Localization. *IEEE Transactions on Image Processing (TIP)*, 30, 2021. Cited on pages 10, 45, 47, 172, and 173.

[KAG22] Bulat Khaertdinov, Stylianos Asteriadis, and Esam Ghaleb. Dynamic Temperature Scaling in Contrastive Self-supervised Learning for Sensor-based Human Activity Recognition. *IEEE Transactions on Biometrics, Behavior, and Identity Science*, 2022. Cited on page 18.

[KB15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. Cited on pages 25, 203, 214, 236, 237, and 258.

[KBS+23] Anna Kukleva, Moritz Böhle, Bernt Schiele, Hilde Kuehne, and Christian Rupprecht. Temperature Schedules for Self-Supervised Contrastive Methods on Long-Tail Data. In *International Conference on Learning Representations (ICLR)*, 2023. Cited on pages 7 and 130.

[KHF+17] Ingo Kilimann, Lucrezia Hausner, Andreas Fellgiebel, Massimo Filippi, Till J Würdemann, Helmut Heinsen, and Stefan J Teipel. Parallel atrophy of cortex and basal forebrain cholinergic system in mild cognitive impairment. *Cerebral Cortex*, 27(3), 2017. Cited on page 36.

[KKHR+14] Yanica Klein-Koerkamp, Rolf Heckemann, Kylee Ramdeen, Olivier Moreaud, Sandrine Keignart, Alexandre Krainik, Alexander Hammers, Monica Baciu, Pascal Hot, and for the Disease Neuroimaging Initiative. Amygdalar Atrophy in Early Alzheimer's Disease. *Current Alzheimer Research*, 11(3), 2014. Cited on page 36.

[KLX+20] Bingyi Kang, Yu Li, Sa Xie, Zehuan Yuan, and Jiashi Feng. Exploring balanced feature spaces for representation learning. In *International Conference on Learning Representations (ICLR)*, 2020. Cited on page 18.

[KMM+20] Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, and Orion Reblitz-Richardson. Captum: A unified and generic model interpretability library for PyTorch. *arXiv:2009.07896*, 2020. Cited on pages 190 and 236.

[KMR+22] Sunnie SY Kim, Nicole Meister, Vikram V Ramaswamy, Ruth Fong, and Olga Russakovsky. HIVE: Evaluating the human interpretability of visual explanations. In *European Conference on Computer Vision (ECCV)*. Springer, 2022. Cited on pages 12, 14, and 142.

[KNT+20] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept Bottleneck Models. In *International Conference on Machine Learning (ICML)*. PMLR, 2020. Cited on page 14.

[Kri09] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. Cited on pages 44, 64, 78, 85, 129, 135, and 191.

[KSA+18] Pieter-Jan Kindermans, Kristof T. Schütt, Maximilian Alber, Klaus-Robert Müller, Dumitru Erhan, Been Kim, and Sven Dähne. Learning how to explain neural networks: PatternNet and PatternAttribution. In *International Conference on Learning Representations (ICLR)*, 2018. Cited on pages 10 and 15.

[KSBD17] Sergey Korolev, Amir Safiullin, Mikhail Belyaev, and Yulia Dodonova. Residual and plain convolutional neural networks for 3D brain MRI classification. In *International Symposium on Biomedical Imaging (ISBI)*. IEEE, 2017. Cited on pages 23, 24, and 25.

[KSC+08] Stefan Klöppel, Cynthia M Stonnington, Carlton Chu, Bogdan Draganski, Rachael I Scahill, Jonathan D Rohrer, Nick C Fox, Clifford R Jack Jr, John Ashburner, and Richard SJ Frackowiak. Automatic classification of MR scans in Alzheimer's disease. *Brain*, 131(3), 2008. Cited on page 23.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems (NeurIPS)*. Curran Associates, Inc., 2012. Cited on page 25.

[KSP+20] Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. Hard negative mixing for contrastive learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. Cited on page 18.

[KTI19] Keisuke Kiritoshi, Ryosuke Tanno, and Tomonori Izumitani. L1-Norm Gradient Penalty for Noise Reduction of Attribution Maps. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Workshop*, 2019. Cited on pages 16 and 49.

[KWG+18] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie J. Cai, James Wexler, Fernanda B. Viégas, and Rory Sayres. Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV). In *International Conference on Machine Learning (ICML)*, 2018. Cited on pages 11 and 12.

[LAV21] Iro Laina, Yuki M Asano, and Andrea Vedaldi. Measuring the Interpretability of Unsupervised Representations via Quantized Reversed Probing. In *International Conference on Learning Representations (ICLR)*, 2021. Cited on page 19.

[LBH15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553), 2015. Cited on page 25.

[LBM+16] Sebastian Lapuschkin, Alexander Binder, Grégoire Montavon, Klaus-Robert Muller, and Wojciech Samek. Analyzing classifiers: Fisher vectors and deep neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. Cited on page 38.

[LCB10] Yann LeCun, Corinna Cortes, and CJ Burges. MNIST handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010. Cited on page 61.

[LCJ+17] Xiaojing Long, Lifang Chen, Chunxiang Jiang, Lijuan Zhang, Alzheimer's Disease Neuroimaging Initiative, et al. Prediction and classification of Alzheimer disease based on quantification of MRI deformation. *PloS one*, 12(3), 2017. Cited on page 36.

[LCWW18] Manhua Liu, Danni Cheng, Kundong Wang, and Yaping Wang. Multi-Modality Cascaded Convolutional Neural Networks for Alzheimer's Disease Diagnosis. *Neuroinformatics*, 2018. Cited on pages 24 and 36.

[LFS21] Max Maria Losch, Mario Fritz, and Bernt Schiele. Semantic bottlenecks: Quantifying and improving inspectability of deep representations. *Int. J. Comput. Vis.*, 129(11), 2021. Cited on page 14.

[LFV20] Iro Laina, Ruth Fong, and Andrea Vedaldi. Quantifying learnability and describability of visual concepts emerging in representation learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 33, 2020. Cited on page 19.

[LH17] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations (ICLR)*, 2017. Cited on page 258.

[LH19] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations (ICLR)*, 2019. Cited on page 120.

[LHGM21] Hong Liu, Jeff Z. HaoChen, Adrien Gaidon, and Tengyu Ma. Self-supervised Learning is More Robust to Dataset Imbalance. In *Advances in Neural Information Processing Systems (NeurIPS), Workshop*, 2021. Cited on page 18.

[Lip18] Zachary C. Lipton. The Mythos of Model Interpretability. *Queue*, 16(3), 2018. Cited on page 37.

[LJE+20] Yuan Liu, Ayush Jain, Clara Eng, David H Way, Kang Lee, Peggy Bui, Kimberly Kanada, Guilherme de Oliveira Marinho, Jessica Gallegos, Sara Gabriele, et al. A deep learning system for differential diagnosis of skin diseases. *Nature medicine*, 26(6), 2020. Cited on page 1.

[LKB+17] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen Awm Van Der Laak, Bram Van Ginneken, and Clara I Sánchez. A survey on deep learning in medical image analysis. *Medical image analysis*, 42, 2017. Cited on pages 23 and 25.

[LL17] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. Cited on page 10.

[LLCR18] Oscar Li, Hao Liu, Chaofan Chen, and Cynthia Rudin. Deep Learning for Case-Based Reasoning through Prototypes: A Neural Network that Explains Its Predictions. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 32, 2018. Cited on page 14.

[LLH+24] Hong Liu, Zhiyuan Li, David Hall, Percy Liang, and Tengyu Ma. Sophia: A Scalable Stochastic Second-order Optimizer for Language Model Pre-training. In *International Conference on Learning Representations (ICLR)*, 2024. Cited on page 142.

[LLLZ20] Tianhong Li, Jianguo Li, Zhuang Liu, and Changshui Zhang. Few sample knowledge distillation for efficient network compression. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. Cited on page 66.

[LLUZ16] Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard Zemel. Understanding the Effective Receptive Field in Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016. Cited on pages 50 and 182.

[LMB+14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *European Conference on Computer Vision (ECCV)*. Springer, 2014. Cited on pages 100, 105, and 236.

[LMW+22] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. Cited on pages 78, 83, 85, 86, 88, and 89.

[LMZ+19] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. Cited on page 18.

[LPD+18] Donghuan Lu, Karteek Popuri, Gavin Weiguang Ding, Rakesh Balachandar, Mirza Faisal Beg, and Alzheimer's Disease Neuroimaging Alzheimer's Disease Neuroimaging Initiative. Multimodal and Multiscale Deep Neural Networks for the Early Diagnosis of Alzheimer's Disease using structural MR and FDG-PET images. *Scientific reports*, 8(1), 2018. Cited on page 24.

[LSD15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. Cited on page 25.

[LSES19] Drew Linsley, Dan Shiebler, Sven Eberhardt, and Thomas Serre. Learning what and where to attend. In *International Conference on Learning Representations (ICLR)*, 2019. Cited on page 16.

[LSG+18] Christian Ledig, Andreas Schuh, Ricardo Guerrero, Rolf A Heckemann, and Daniel Rueckert. Structural brain imaging in Alzheimer's disease and mild cognitive impairment: biomarker analysis and shared morphometry database. *Scientific reports*, 8(1), 2018. Cited on page 36.

[LWP+18] Kunpeng Li, Ziyan Wu, Kuan-Chuan Peng, Jan Ernst, and Yun Fu. Tell Me Where to Look: Guided Attention Inference Network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. Cited on pages 16 and 17.

[LWWB19] Boyi Li, Felix Wu, Kilian Q Weinberger, and Serge Belongie. Positional normalization. *Advances in Neural Information Processing Systems (NeurIPS)*, 32, 2019. Cited on page 83.

[Mat21] Yoshitomo Matsubara. torchdistill: A Modular, Configuration-Driven Framework for Knowledge Distillation. In *International Workshop on Reproducible Research in Pattern Recognition*. Springer, 2021. Cited on page 255.

[MBL+19] Grégoire Montavon, Alexander Binder, Sebastian Lapuschkin, Wojciech Samek, and Klaus-Robert Müller. Layer-Wise Relevance Propagation: An Overview. *Explainable AI: interpreting, explaining and visualizing deep learning*, 2019. Cited on pages 10, 45, and 52.

[mc16] TorchVision maintainers and contributors. TorchVision: PyTorch's Computer Vision library. https://github.com/pytorch/vision, 2016. Cited on pages 236 and 258.

[MFBS22] Ofir Moshe, Gil Fidel, Ron Bitton, and Asaf Shabtai. Improving Interpretability via Regularization of Neural Activation Sensitivity. *arXiv:2211.08686*, 2022. Cited on page 16.

[MFS+21] Masahiro Mitsuhara, Hiroshi Fukui, Yusuke Sakashita, Takanori Ogata, Tsubasa Hirakawa, Takayoshi Yamashita, and Hironobu Fujiyoshi. Embedding Human Knowledge into Deep Neural Network via Attention Map. In *VISIGRAPP*, 2021. Cited on page 16.

[MGRR+11] Melissa E Murray, Neill R Graff-Radford, Owen A Ross, Ronald C Petersen, Ranjan Duara, and Dennis W Dickson. Neuropathologically defined subtypes of Alzheimer's disease with distinct clinical characteristics: a retrospective study. *The Lancet Neurology*, 10(9), 2011. Cited on page 37.

[MH21] Samuel G. Müller and Frank Hutter. TrivialAugment: Tuning-free Yet State-of-the-Art Data Augmentation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. Cited on page 215.

[MKSFG20] Sreyas Mohan, Zahra Kadkhodaie, Eero P Simoncelli, and Carlos Fernandez-Granda. Robust And Interpretable Blind Image Denoising Via Bias-Free Convolutional Neural Networks. In *International Conference on Learning Representations (ICLR)*, 2020. Cited on page 84.

[MLB+17] Grégoire Montavon, Sebastian Lapuschkin, Alexander Binder, Wojciech Samek, and Klaus-Robert Müller. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition*, 65, 2017. Cited on page 45.

[Mor93] John C. Morris. The Clinical Dementia Rating (CDR). *Neurology*, 43(11), 1993. Cited on page 25.

[MPCB14] Guido F. Montúfar, Razvan Pascanu, KyungHyun Cho, and Yoshua Bengio. On the Number of Linear Regions of Deep Neural Networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014. Cited on page 57.

[MSK+19] W James Murdoch, Chandan Singh, Karl Kumbier, Reza Abbasi-Asl, and Bin Yu. Definitions, methods, and applications in interpretable machine learning. *Proceedings of the National Academy of Sciences*, 116(44), 2019. Cited on pages 2 and 3.

[MSM18] Grégoire Montavon, Wojciech Samek, and Klaus Robert Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73, 2018. Cited on pages 26 and 38.

[MV23] Sachit Menon and Carl Vondrick. Visual Classification via Description from Large Language Models. In *International Conference on Learning Representations (ICLR)*, 2023. Cited on page 14.

[NH10] Vinod Nair and Geoffrey E. Hinton. Rectified Linear Units Improve Restricted Boltzmann Machines. In *International Conference on Machine Learning (ICML)*, 2010. Cited on pages 4, 15, 57, 83, and 102.

[NJL+14] Young Noh, Seun Jeon, Jong Min Lee, Sang Won Seo, Geon Ha Kim, Hanna Cho, Byoung Seok Ye, Cindy W Yoon, Hee Jin Kim, Juhee Chin, et al. Anatomical heterogeneity of Alzheimer disease based on cortical thickness on MRIs. *Neurology*, 83(21), 2014. Cited on page 37.

[NS20] Krishna Kanth Nakka and Mathieu Salzmann. Towards Robust Fine-Grained Recognition by Maximal Separation of Discriminative Features. In *ACCV*, 2020. Cited on page 16.

[NTP+23] Meike Nauta, Jan Trienes, Shreyasi Pathak, Elisa Nguyen, Michelle Peters, Yasmin Schmitt, Jörg Schlötterer, Maurice van Keulen, and Christin Seifert. From anecdotal evidence to quantitative evaluation methods: A systematic review on evaluating explainable ai. *ACM Computing Surveys*, 55(13s), 2023. Cited on pages 2, 12, 13, and 139.

[NVBS21] Meike Nauta, Ron Van Bree, and Christin Seifert. Neural Prototype Trees for Interpretable Fine-grained Image Recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. Cited on page 14.

[OCS+20] Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. *Distill*, 5(3), 2020. Cited on page 11.

[ODNW23] Tuomas Oikarinen, Subhro Das, Lam M. Nguyen, and Tsui-Wei Weng. Label-free Concept Bottleneck Models. In *International Conference on Learning Representations (ICLR)*, 2023. Cited on page 14.

[OLL22] Utkarsh Ojha, Yuheng Li, and Yong Jae Lee. What Knowledge gets Distilled in Knowledge Distillation? *arXiv:2205.16004*, 2022. Cited on page 17.

[OLV18] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv:1807.03748*, 2018. Cited on pages 18, 19, and 131.

[OMS17] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2(11), 2017. Cited on page 11.

[OPYM+12] Graziella Orrù, William Pettersson-Yeo, Andre F Marquand, Giuseppe Sartori, and Andrea Mechelli. Using support vector machine to identify imaging biomarkers of neurological and psychiatric disease: a critical review. *Neuroscience & Biobehavioral Reviews*, 36, 2012. Cited on page 23.

[OWJ+22] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems (NeurIPS)*, 35, 2022. Cited on pages 139 and 142.

[PABRS23] Amin Parchami-Araghi, Moritz Böhle, Sukrut Rao, and Bernt Schiele. Good Teachers Explain: Explanaiton-enhanced Knowledge Distillation. 2023. Under review. Cited on page 7.

[Par94] Vilfredo Pareto. Il massimo di utilità dato dalla libera concorrenza. *Giornale degli economisti*, 1894. Cited on page 105.

[Par08] Vilfredo Pareto. The Maximum of Utility given by Free Competition. *Giornale degli Economisti e Annali di Economia*, 67(3), 2008. Cited on page 105.

[PDN+22] Suzanne Petryk, Lisa Dunlap, Keyan Nasseri, Joseph Gonzalez, Trevor Darrell, and Anna Rohrbach. On Guiding Visual Attention with Language Specification. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. Cited on pages 16, 99, 101, 105, 112, 116, 233, 237, 250, 256, and 259.

[PDS18] Vitali Petsiuk, Abir Das, and Kate Saenko. RISE: Randomized Input Sampling for Explanation of Black-box Models. In *British Machine Vision Conference (BMVC)*, 2018. Cited on pages 10, 13, 45, 67, 71, 102, 174, 190, and 205.

[PGM+19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. Cited on pages 86, 190, 203, 204, 236, and 258.

[PIM23] Konstantinos Panagiotis Panousis, Dino Ienco, and Diego Marcos. Sparse linear concept discovery models. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. Cited on page 14.

[PKO+22] Vipin Pillai, Soroush Abbasi Koohpayegani, Ashley Ouligian, Dennis Fong, and Hamed Pirsiavash. Consistent Explanations by Contrastive Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. Cited on page 16.

[PKR+19] Kristina Preuer, Günter Klambauer, Friedrich Rippmann, Sepp Hochreiter, and Thomas Unterthiner. Interpretable deep learning in drug discovery. *Explainable AI: interpreting, explaining and visualizing deep learning*, 2019. Cited on page 3.

[PM15] Adrien Payan and Giovanni Montana. Predicting Alzheimer's disease: a neuroimaging study with 3D convolutional neural networks. *arXiv:1502.02506*, 2015. Cited on pages 23 and 25.

[PNK+17] Jong-Yun Park, Han Kyu Na, Sungsoo Kim, Hyunwook Kim, Hee Jin Kim, Sang Won Seo, Duk L Na, Cheol E Han, Joon-Kyung Seong, Michael Weiner, et al. Robust identification of Alzheimer's disease subtypes based on cortical atrophy patterns. *Scientific reports*, 7, 2017. Cited on pages 30, 36, and 37.

[PP21] Vipin Pillai and Hamed Pirsiavash. Explainable Models with Consistent Interpretations. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2021. Cited on page 16.

[QSR+13] Y. T. Quiroz, C. E. Stern, E. M. Reiman, M. Brickhouse, A. Ruiz, R. A. Sperling, F. Lopera, and B. C. Dickerson. Cortical atrophy in presymptomatic Alzheimer's disease presenilin 1 mutation carriers. *Journal of Neurology, Neurosurgery & Psychiatry*, 84(5), 2013. Cited on page 36.

[RBK+15] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. FitNets: Hints for Thin Deep Nets. In *International Conference on Learning Representations (ICLR)*, 2015. Cited on page 17.

[RBPAS23] Sukrut Rao, Moritz Böhle, Amin Parchami-Araghi, and Bernt Schiele. Studying How to Efficiently and Effectively Guide Models with Explanations. *ICCV*, 2023. Cited on pages 6, 7, 101, 118, 120, 123, 259, and 260.

[RBS22] Sukrut Rao, Moritz Böhle, and Bernt Schiele. Towards Better Understanding Attribution Methods. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. Cited on pages 5, 40, 90, 102, and 227.

[RBS24] Sukrut Rao, Moritz Böhle, and Bernt Schiele. Better Understanding Differences in Attribution Methods via Systematic Evaluations. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2024. Cited on pages 5 and 40.

[RCC+22] Cynthia Rudin, Chaofan Chen, Zhi Chen, Haiyang Huang, Lesia Semenova, and Chudi Zhong. Interpretable machine learning: Fundamental principles and 10 grand challenges. *Statistic Surveys*, 16, 2022. Cited on pages 4 and 139.

[RCSJ20] Joshua Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. Contrastive learning with hard negative samples. In *International Conference on Learning Representations (ICLR)*, 2020. Cited on page 18.

[Ree01] William J Reed. The Pareto, Zipf and other power laws. *Economics Letters*, 2001. Cited on pages 18 and 129.

[REW+18] Johannes Rieke, Fabian Eitel, Martin Weygandt, John-Dylan Haynes, and Kerstin Ritter. Visualizing Convolutional Networks for MRI-Based Diagnosis of Alzheimer's Disease. In *Understanding and Interpreting Machine Learning in Medical Image Computing Applications*. Springer, 2018. Cited on pages 24 and 27.

[RHDV17] Andrew Slavin Ross, Michael C Hughes, and Finale Doshi-Velez. Right for the Right Reasons: Training Differentiable Models by Constraining their Explanations. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2017. Cited on pages 16, 17, 100, 102, 104, 105, 115, and 118.

[RHI+17] Saima Rathore, Mohamad Habes, Muhammad Aksam Iftikhar, Amanda Shacklett, and Christos Davatzikos. A review on neuroimaging-based classification studies and associated feature extraction methods for Alzheimer's disease and its prodromal stages. *NeuroImage*, 155, 2017. Cited on page 23.

[RKH+21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models from Natural Language Supervision. In *International Conference on Machine Learning (ICML)*, 2021. Cited on pages 118, 139, and 142.

[RLW+16] Kerstin Ritter, Catharina Lange, Martin Weygandt, Anja Mäurer, Anna Roberts, Melanie Estrella, Per Suppa, Lothar Spies, Vikas Prasad, Ingo Steffen, and Others. Combination of Structural MRI and FDG-PET of the Brain Improves Diagnostic Accuracy in Newly Manifested Cognitive Impairment in Geriatric Inpatients. *Journal of Alzheimer's Disease*, 54(4), 2016. Cited on page 23.

[RSG16] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Conference on Knowledge Discovery and Data Mining (KDD)*, 2016. Cited on pages 10, 12, 67, 90, 102, 205, and 215.

[RSMY20] Laura Rieger, Chandan Singh, William Murdoch, and Bin Yu. Interpretations are Useful: Penalizing Explanations to Align Neural Networks with Prior Knowledge. In *International Conference on Machine Learning (ICML)*, 2020. Cited on pages 16 and 17.

[RSW+15] Kerstin Ritter, Julia Schumacher, Martin Weygandt, Ralph Buchert, Carsten Allefeld, and John-Dylan Haynes. Multimodal prediction of conversion to Alzheimer's disease based on incomplete biomarkers. *Alzheimer's & Dementia: Diagnosis, Assessment & Disease Monitoring*, 1(2), 2015. Cited on page 23.

[RSY+21] Joshua David Robinson, Li Sun, Ke Yu, kayhan Batmanghelich, Stefanie Jegelka, and Suvrit Sra. Can contrastive learning avoid shortcut solutions? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. Cited on pages 19, 131, and 133.

[Rud19] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5), 2019. Cited on pages 1, 3, 4, and 139.

[RZL18] Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for Activation Functions. In *International Conference on Learning Representations (ICLR), Workshop*, 2018. Cited on page 79.

[SBM+16] Wojciech Samek, Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, and Klaus-Robert Müller. Evaluating the visualization of what a Deep Neural Network has learned. *IEEE Transactions on Neural Networks and Learning Systems*, 28(11), 2016. Cited on pages 12, 13, 24, 37, and 42.

[SCD+17] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017. Cited on pages 10, 16, 43, 44, 45, 67, 90, 100, 102, 105, 115, 117, 120, 174, 179, 182, 192, 205, 215, 227, and 250.

[SCR+22] Nina Shvetsova, Brian Chen, Andrew Rouditchenko, Samuel Thomas, Brian Kingsbury, Rogerio S Feris, David Harwath, James Glass, and Hilde Kuehne. Everything at Once-Multi-Modal Fusion Transformer for Video Retrieval. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. Cited on page 18.

[SDBR15] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for Simplicity: The All Convolutional Net. In *International Conference on Learning Representations (ICLR), Workshop*, 2015. Cited on pages 10, 24, 27, 45, 90, 173, 174, 190, and 215.

[SF18] Suraj Srinivas and François Fleuret. Knowledge Transfer with Jacobian Matching. In *International Conference on Machine Learning (ICML)*, 2018. Cited on page 17.

[SF19] Suraj Srinivas and François Fleuret. Full-Gradient Representation for Neural Network Visualization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. Cited on pages 10, 12, 13, 42, 67, 68, 84, 92, and 205.

[SFH17] Sara Sabour, Nicholas Frosst, and Geoffrey E. Hinton. Dynamic Routing Between Capsules. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. Cited on pages 15, 60, 194, 208, and 209.

[SGGP+16] Nienke ME Scheltens, Francisca Galindo-Garre, Yolande AL Pijnenburg, Annelies E van der Vlies, Lieke L Smits, Teddy Koene, Charlotte E Teunissen, Frederik Barkhof, Mike P Wattjes, Philip Scheltens, et al. The identification of cognitive subtypes in Alzheimer's disease dementia using latent class analysis. *Journal of Neurology, Neurosurgery, and Psychiatry*, 87(3), 2016. Cited on page 37.

[SGK17] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning Important Features Through Propagating Activation Differences. In *International Conference on Machine Learning (ICML)*, 2017. Cited on pages 10, 15, 45, 52, 67, 90, 100, 102, 105, 173, 179, 182, 187, 190, 192, 205, and 215.

[SGTM19] Kristof T Schütt, Michael Gastegger, Alexandre Tkatchenko, and Klaus-Robert Müller. Quantum-chemical insights from interpretable atomistic neural networks. *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, 2019. Cited on page 3.

[SH20] Hua Shen and Ting-Hao Huang. How Useful Are the Machine-Generated Interpretations to General Users? A Human Evaluation on Guessing the Incorrectly Predicted Labels. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 8, 2020. Cited on page 12.

[Sha18] Irhum Shafkat. Intuitively Understanding Convolutions for Deep Learning. https://towardsdatascience.com/intuitively-understanding-convolutions-for-deep-learning-1f6f42faee1#ad33, 2018. Cited on page 62.

[SIK+21] Samuel Stanton, Pavel Izmailov, Polina Kirichenko, Alexander A Alemi, and Andrew G Wilson. Does Knowledge Distillation Really Work? In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34. Curran Associates, Inc., 2021. Cited on pages 17, 115, and 118.

[SJN21] Harshay Shah, Prateek Jain, and Praneeth Netrapalli. Do Input Gradients Highlight Discriminative Features? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. Cited on page 13.

[SKHL20] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally Robust Neural Networks for Group Shifts: On the Importance of Regularization for Worst-Case Generalization. In *International Conference on Learning Representations (ICLR)*, 2020. Cited on pages 17, 101, 105, 112, 116, 118, 120, 233, 237, 250, 256, and 259.

[SKL+20] Guolei Sun, Salman Khan, Wen Li, Hisham Cholakkal, Fahad Shahbaz Khan, and Luc Van Gool. Fixing Localization Errors to Improve Image Classification. In *European Conference on Computer Vision (ECCV)*. Springer, 2020. Cited on page 16.

[SLJ+15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going Deeper with Convolutions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. Cited on pages 47 and 172.

[SLL+21] Haifeng Shen, Kewen Liao, Zhibin Liao, Job Doornberg, Maoying Qiao, Anton Van Den Hengel, and Johan W Verjans. Human-AI Interactive and Continuous Sensemaking: A Case Study of Image Classification using Scribble Attention Maps. In *Extended Abstracts of CHI*, 2021. Cited on pages 16, 100, 102, 104, and 105.

[SLS14] Heung-Il Suk, Seong-Whan Lee, and Dinggang Shen. Hierarchical feature representation and multimodal fusion with deep learning for AD/MCI diagnosis. *NeuroImage*, 101, 2014. Cited on pages 23 and 25.

[SLS+19] Ramprasaath R Selvaraju, Stefan Lee, Yilin Shen, Hongxia Jin, Shalini Ghosh, Larry Heck, Dhruv Batra, and Devi Parikh. Taking a HINT: Leveraging Explanations to Make Vision and Language Models More Grounded. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. Cited on page 16.

[SLSM16] Irene Sturm, Sebastian Lapuschkin, Wojciech Samek, and Klaus-Robert Müller. Interpretable deep neural networks for single-trial EEG classification. *Journal of neuroscience methods*, 274, 2016. Cited on page 24.

[SMG+20] Krishna Kumar Singh, Dhruv Mahajan, Kristen Grauman, Yong Jae Lee, Matt Feiszli, and Deepti Ghadiyaram. Don't Judge an Object by its Context: Learning to Overcome Contextual Bias. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. Cited on page 16.

[SML+21] Wojciech Samek, Grégoire Montavon, Sebastian Lapuschkin, Christopher J Anders, and Klaus-Robert Müller. Explaining deep neural networks and beyond: A review of methods and applications. *Proceedings of the IEEE*, 109(3), 2021. Cited on pages 3 and 141.

[SPA+19] Nikunj Saunshi, Orestis Plevrakis, Sanjeev Arora, Mikhail Khodak, and Hrishikesh Khandeparkar. A theoretical analysis of contrastive unsupervised representation learning. In *International Conference on Machine Learning (ICML)*, 2019. Cited on page 19.

[SPA+22] Steven Stalder, Nathanaël Perraudin, Radhakrishna Achanta, Fernando Perez-Cruz, and Michele Volpi. What You See is What You Classify: Black Box Attributions. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. Cited on page 236.

[SRS+23] Mikołaj Sacha, Dawid Rymarczyk, Łukasz Struski, Jacek Tabor, and Bartosz Zieliński. ProtoSeg: Interpretable Semantic Segmentation with Prototypical Parts. In *Winter Conference on Applications of Computer Vision (WACV)*, 2023. Cited on page 14.

[SSP+22] Leon Sixt, Martin Schuessler, Oana-Iuliana Popescu, Philipp Weiß, and Tim Landgraf. Do Users Benefit From Interpretable Vision? A User Study, Baseline, And Dataset. In *International Conference on Learning Representations (ICLR)*, 2022. Cited on page 12.

[SSPP20] Aniruddha Saha, Akshayvarun Subramanya, Koninika Patil, and Hamed Pirsiavash. Role of Spatial Context in Adversarial Robustness for Object Detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Workshop*, 2020. Cited on page 46.

[SSS+21] Xiaoting Shao, Arseny Skryagin, Wolfgang Stammer, Patrick Schramowski, and Kristian Kersting. Right for Better Reasons: Training Differentiable Models by Constraining their Influence Functions. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 35, 2021. Cited on page 16.

[SST+20] Patrick Schramowski, Wolfgang Stammer, Stefano Teso, Anna Brugger, Franziska Herbert, Xiaoting Shao, Hans-Georg Luigs, Anne-Katrin Mahlein, and Kristian Kersting. Making Deep Neural Networks Right for the Right Scientific Reasons by Interacting with their Explanations. *Nature Machine Intelligence*, 2(8), 2020. Cited on page 16.

[SSTL20] Karl Schulz, Leon Sixt, Federico Tombari, and Tim Landgraf. Restricting the flow: Information bottlenecks for attribution. *arXiv:2001.00396*, 2020. Cited on pages 12 and 13.

[ST16] Saman Sarraf and Ghassem Tofighi. DeepAD: Alzheimer's Disease Classification via Deep Convolutional Neural Networks using MRI and fMRI. *bioRxiv*, 2016. Cited on pages 23, 24, and 25.

[STK+17] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv:1706.03825*, 2017. Cited on pages 49 and 187.

[STY17] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic Attribution for Deep Networks. In *International Conference on Machine Learning (ICML)*, 2017. Cited on pages 10, 45, 46, 52, 67, 84, 90, 100, 102, 105, 174, 179, 182, 187, 189, 190, 205, and 215.

[Sun16] Lei Sun. ResNet on Tiny ImageNet. http://cs231n.stanford.edu/reports/2017/pdfs/12.pdf, 2016. Accessed: 2020-11-16. Cited on page 66.

[SVZ14] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. In *International Conference on Learning Representations (ICLR), Workshop*, 2014. Cited on pages 10, 45, 173, 174, and 190.

[SX22] Zhiqiang Shen and Eric Xing. A Fast Knowledge Distillation Framework for Visual Recognition. In *European Conference on Computer Vision (ECCV)*, Cham, 2022. Springer Nature Switzerland. Cited on pages 18 and 119.

[SZ15] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *International Conference on Learning Representations (ICLR)*, 2015. Cited on pages 44, 47, 59, 66, 78, 86, 172, 173, 179, 182, 184, and 191.

[TASD23] Stefano Teso, Öznur Alkan, Wolfgang Stammer, and Elizabeth Daly. Leveraging Explanations in Interactive Machine Learning: An Overview. *Frontiers in Artificial Intelligence*, 6, 2023. Cited on page 16.

[TAvdH20] Damien Teney, Ehsan Abbasnedjad, and Anton van den Hengel. Learning What Makes a Difference from Counterfactual Examples and Gradient Supervision. In *European Conference on Computer Vision (ECCV)*. Springer, 2020. Cited on page 16.

[Tes19] Stefano Teso. Toward Faithful Explanatory Active Learning with Self-Explainable Neural Nets. In *Workshop on IAL*. CEUR Workshop Proceedings, 2019. Cited on pages 16 and 100.

[THMS18] Armin W Thomas, Hauke R Heekeren, Klaus-Robert Müller, and Wojciech Samek. Interpretable LSTMs For Whole-Brain Neuroimaging Analyses. *arXiv:1810.09945*, 2018. Cited on page 24.

[THvdO21] Yonglong Tian, Olivier J Henaff, and Aäron van den Oord. Divide and contrast: Self-supervised learning from uncurated data. In *ICCV*, 2021. Cited on page 18.

[TJM20] Chakkrit Termritthikun, Yeshi Jamtsho, and Paisarn Muneesawang. An improved residual network model for image recognition using a combination of snapshot ensembles and the cutout technique. *Multimedia Tools and Applications*, 2020. Cited on page 66.

[TK19] Stefano Teso and Kristian Kersting. Explanatory Interactive Machine Learning. In *AIES*, 2019. Cited on pages 16 and 100.

[TKI20] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *European Conference on Computer Vision (ECCV)*, 2020. Cited on pages 18 and 135.

[Tor] Torchvision library, pretrained models. https://pytorch.org/vision/stable/models.html. Accessed: 2021-11-11. Cited on pages 86, 88, and 89.

[Tor21] How to Train State-Of-The-Art Models Using TorchVision's Latest Primitives. https://pytorch.org/blog/how-to-train-state-of-the-art-models-using-torchvision-latest-primitives/, 2021. Accessed: 2023-05-23. Cited on pages 86 and 89.

[TSE+19] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness May Be at Odds with Accuracy. In *International Conference on Learning Representations (ICLR)*, 2019. Cited on page 3.

[TSP+20] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning? *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. Cited on page 265.

[TWSM20] Yao-Hung Hubert Tsai, Yue Wu, Ruslan Salakhutdinov, and Louis-Philippe Morency. Self-supervised learning from a multi-view perspective. In *International Conference on Learning Representations (ICLR)*, 2020. Cited on page 19.

[UVL16] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv:1607.08022*, 2016. Cited on page 83.

[VGVGVG21] Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, and Luc Van Gool. Unsupervised semantic segmentation by contrasting object mask proposals. In *ICCV*, 2021. Cited on page 18.

[VPM17] Sandra Vieira, Walter H.L. Pinaya, and Andrea Mechelli. Using deep learning to investigate the neuroimaging correlates of psychiatric and neurological disorders: Methods and applications. *Neuroscience & Biobehavioral Reviews*, 74, 2017. Cited on page 25.

[VPW+13] Latha Velayudhan, Petroula Proitsi, Eric Westman, J-Sebastian Muehlboeck, Patrizia Mecocci, Bruno Vellas, Magda Tsolaki, Iwona Kłoszewska, Hilkka Soininen, Christian Spenger, Angela Hodges, John Powell, Simon Lovestone, Andrew Simmons, and dNeuroMed Consortium. Entorhinal Cortex Thickness Predicts Cognitive Decline in Alzheimer's Disease. *Journal of Alzheimer's Disease*, 33(3), 2013. Cited on page 36.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. Cited on pages 15 and 85.

[WDH+21] Alvin Wan, Lisa Dunlap, Daniel Ho, Jihan Yin, Scott Lee, Suzanne Petryk, Sarah Adel Bargal, and Joseph E. Gonzalez. NBDT: Neural-Backed Decision Tree. In *International Conference on Learning Representations (ICLR)*, 2021. Cited on page 14.

[WFX+22] Chen Wei, Haoqi Fan, Saining Xie, Chao-Yuan Wu, Alan Yuille, and Christoph Feichtenhofer. Masked feature prediction for self-supervised visual pre-training. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. Cited on page 18.

[WH18] Yuxin Wu and Kaiming He. Group normalization. In *European Conference on Computer Vision (ECCV)*, 2018. Cited on page 83.

[WHO17] WHO. Dementia. *Available at: https://www.who.int/news-room/fact-sheets/detail/dementia*, 2017. Cited on page 23.

[WI20] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning (ICML)*, 2020. Cited on pages 19 and 131.

[WL21a] Feng Wang and Huaping Liu. Understanding the behaviour of contrastive loss. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. Cited on pages 19, 131, 132, and 265.

[WL21b] Zixin Wen and Yuanzhi Li. Toward understanding the feature learning process of self-supervised contrastive learning. In *International Conference on Machine Learning (ICML)*, 2021. Cited on page 19.

[WLWJ21] Jiaqi Wang, Huafeng Liu, Xinyue Wang, and Liping Jing. Interpretable Image Recognition by Constructing Transparent Embedding Space. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. Cited on page 14.

[WRH17] Yu-Xiong Wang, Deva Ramanan, and Martial Hebert. Learning to model the tail. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. Cited on page 18.

[WVA+13] Michael W Weiner, Dallas P Veitch, Paul S Aisen, Laurel A Beckett, Nigel J Cairns, Robert C Green, Danielle Harvey, Clifford R Jack, William Jagust, Enchi Liu, John C Morris, Ronald C Petersen, Andrew J Saykin, Mark E Schmidt, Leslie Shaw, Li Shen, Judith A Siuciak, Holly Soares, Arthur W Toga, John Q Trojanowski, and The Alzheimer's Disease Neuroimaging Initiative. The Alzheimer's Disease Neuroimaging Initiative: A review of papers published since its inception. *Alzheimer's & Dementia*, 9, 2013. Cited on pages 23 and 36.

[WWD+20] Haofan Wang, Zifan Wang, Mengnan Du, Fan Yang, Zijian Zhang, Sirui Ding, Piotr Mardziel, and Xia Hu. Score-CAM: Score-Weighted Visual Explanations for Convolutional Neural Networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Workshop*, 2020. Cited on pages 10, 16, 17, 45, 100, 101, 103, and 104.

[WWW+21] Guangrun Wang, Keze Wang, Guangcong Wang, Philip HS Torr, and Liang Lin. Solving inefficiency of self-supervised representation learning. In *ICCV*, 2021. Cited on page 18.

[WXSL18] Zhirong Wu, Yuanjun Xiong, X Yu Stella, and Dahua Lin. Unsupervised Feature Learning via Non-Parametric Instance Discrimination. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. Cited on pages 18 and 131.

[XEIM21] Kai Yuanqing Xiao, Logan Engstrom, Andrew Ilyas, and Aleksander Madry. Noise or Signal: The Role of Image Backgrounds in Object Recognition. In *International Conference on Learning Representations (ICLR)*, 2021. Cited on page 99.

[XGD+17] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. Cited on pages 47, 78, 86, and 172.

[XHE+10] Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. SUN database: Large-Scale Scene Recognition from Abbey to Zoo. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. Cited on pages 120 and 123.

[XSM+21] Tete Xiao, Mannat Singh, Eric Mintun, Trevor Darrell, Piotr Dollár, and Ross Girshick. Early convolutions help transformers see better. *Advances in Neural Information Processing Systems (NeurIPS)*, 34, 2021. Cited on pages 86 and 90.

[YBZ+22] Xufeng Yao, Yang Bai, Xinyun Zhang, Yuechen Zhang, Qi Sun, Ran Chen, Ruiyu Li, and Bei Yu. PCL: Proxy-based Contrastive Learning for Domain Generalization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. Cited on page 18.

[YHH+21] Chun-Hsiao Yeh, Cheng-Yao Hong, Yen-Chi Hsu, Tyng-Luh Liu, Yubei Chen, and Yann LeCun. Decoupled contrastive learning. *arXiv:2110.06848*, 2021. Cited on page 18.

[YHO+19] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon 5 Yoo. CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. Cited on page 214.

[YK19] Mengjiao Yang and Been Kim. Benchmarking Attribution Methods with Relative Feature Importance. *CoRR*, abs/1907.09701, 2019. Cited on page 12.

[YKDO23] Ziyan Yang, Kushal Kafle, Franck Dernoncourt, and Vicente Ordonez. Improving Visual Grounding by Encouraging Consistent Gradient-based Explanations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. Cited on page 16.

[YOH+21] Sangdoo Yun, Seong Joon Oh, Byeongho Heo, Dongyoon Han, Junsuk Choe, and Sanghyuk Chun. Re-Labeling ImageNet: From Single to Multi-Labels, From Global to Localized Labels. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. Cited on pages 18 and 119.

[YRR18] Chengliang Yang, Anand Rangarajan, and Sanjay Ranka. Visual Explanations From Deep 3D Convolutional Neural Networks for Alzheimer's Disease Classification. *arXiv:1803.02544*, 2018. Cited on page 24.

[YWZ23] Mert Yuksekgonul, Maggie Wang, and James Zou. Post-hoc Concept Bottleneck Models. In *International Conference on Learning Representations (ICLR)*, 2023. Cited on page 14.

[YX20] Yuzhe Yang and Zhi Xu. Rethinking the value of labels for improving class-imbalanced learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. Cited on page 18.

[YXK+21] Sijie Yan, Yuanjun Xiong, Kaustav Kundu, Shuo Yang, Siqi Deng, Meng Wang, Wei Xia, and Stefano Soatto. Positive-congruent Training: Towards Regression-free Model Updates. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. Cited on page 115.

[YXZ+23] Penghui Yang, Ming-Kun Xie, Chen-Chen Zong, Lei Feng, Gang Niu, Masashi Sugiyama, and Sheng-Jun Huang. Multi-Label Knowledge Distillation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. Cited on pages 120, 123, and 259.

[ZBL+18] Jianming Zhang, Sarah Adel Bargal, Zhe Lin, Jonathan Brandt, Xiaohui Shen, and Stan Sclaroff. Top-Down Neural Attention by Excitation Backprop. *International Journal of Computer Vision (IJCV)*, 126(10), 2018. Cited on pages 10, 12, 13, 52, 67, and 205.

[ZCDLP18] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond Empirical Risk Minimization. In *International Conference on Learning Representations (ICLR)*, 2018. Cited on pages 122 and 214.

[ZCS+22] Borui Zhao, Quan Cui, Renjie Song, Yiyu Qiu, and Jiajun Liang. Decoupled knowledge distillation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. Cited on page 17.

[ZF14] Matthew D Zeiler and Rob Fergus. Visualizing and Understanding Convolutional Networks. In *European Conference on Computer Vision (ECCV)*, 2014. Cited on pages 10, 45, 67, 174, 179, 190, 192, and 205.

[ZHS18] Pourya Habib Zadeh, Reshad Hosseini, and Suvrit Sra. Deep-RBF Networks Revisited: Robust Classification with Rejection. *ArXiv*, abs/1812.03190, 2018. Cited on pages 16, 143, and 216.

[ZJM+21] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning (ICML)*, 2021. Cited on page 18.

[ZK16] Sergey Zagoruyko and Nikos Komodakis. Wide Residual Networks. In *British Machine Vision Conference (BMVC)*, 2016. Cited on pages 47, 66, and 172.

[ZK17] Sergey Zagoruyko and Nikos Komodakis. Paying More Attention to Attention: Improving the Performance of Convolutional Neural Networks via Attention Transfer. In *International Conference on Learning Representations (ICLR)*, 2017. Cited on pages 17, 120, 121, and 255.

[ZKB23] Roland S. Zimmermann, Thomas Klein, and Wieland Brendel. Scale Alone Does not Improve Mechanistic Interpretability in Vision Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. Cited on page 139.

[ZKL+15] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Object Detectors Emerge in Deep Scene CNNs. In *ICLR*, 2015. Cited on page 11.

[ZKL+16] Bolei Zhou, Aditya Khosla, Àgata Lapedriza, Aude Oliva, and Antonio Torralba. Learning Deep Features for Discriminative Localization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. Cited on pages 2, 10, and 17.

[ZMM+21] Ruihan Zhang, Prashan Madumal, Tim Miller, Krista A Ehinger, and Benjamin IP Rubinstein. Invertible concept-based explanations for cnn models with non-negative concept activation vectors. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 35, 2021. Cited on page 11.

[ZMS+16] Xiuming Zhang, Elizabeth C Mormino, Nanbo Sun, Reisa A Sperling, Mert R Sabuncu, B T Thomas Yeo, and the Alzheimer's Disease Neuroimaging Alzheimer's Disease Neuroimaging Initiative. Bayesian model reveals latent atrophy factors with dissociable cognitive trajectories in Alzheimer's disease. *Proceedings of the National Academy of Sciences of the United States of America*, 113(42), 2016. Cited on page 37.

[ZSZ+22] Michael Zhang, Nimit Sharad Sohoni, Hongyang R. Zhang, Chelsea Finn, and Christopher Ré. Correct-N-Contrast: a Contrastive Approach for Improving Robustness to Spurious Correlations. In *International Conference on Machine Learning (ICML)*, 2022. Cited on page 16.

[ZTC+22] Yuanyi Zhong, Haoran Tang, Junkun Chen, Jian Peng, and Yu-Xiong Wang. Is Self-Supervised Learning More Robust Than Supervised Learning? *arXiv:2206.05259*, 2022. Cited on page 18.

[ZWBG21] Oliver Zhang, Mike Wu, Jasmine Bayrooti, and Noah Goodman. Temperature as Uncertainty in Contrastive Learning. *arXiv:2110.04403*, 2021. Cited on page 265.

[ZYW+22] Zhihan Zhou, Jiangchao Yao, Yan-Feng Wang, Bo Han, and Ya Zhang. Contrastive learning with boosted memorization. In *International Conference on Machine Learning (ICML)*. PMLR, 2022. Cited on page 19.

[ZZK+20] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random Erasing Data Augmentation. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2020. Cited on page 215.

[ZZP+22] Chaoning Zhang, Kang Zhang, Trung X Pham, Axi Niu, Zhinan Qiao, Chang D Yoo, and In So Kweon. Dual temperature helps contrastive learning without many negative samples: Towards understanding and simplifying moco. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. Cited on pages 18 and 265.

APPENDIX — UNDERSTANDING ATTRIBUTIONS

In this supplement to our work on understanding attribution methods, we provide:

A.1 Quantitative Results on All Models

In Figure A.1, we provide the full results of our quantitative evaluation on the Grid Pointing Game [BFS21] (GridPG), DiFull, and DiPart using the backpropagation-based (left), activation-based (middle), and perturbation-based (right) methods on VGG-19 [SZ15], ResNet-152 [HZRS16], VGG-11 [SZ15], ResNet-18 [HZRS16], Wide ResNet [ZK16], Resnext [XGD+17], and Googlenet [SLJ+15].

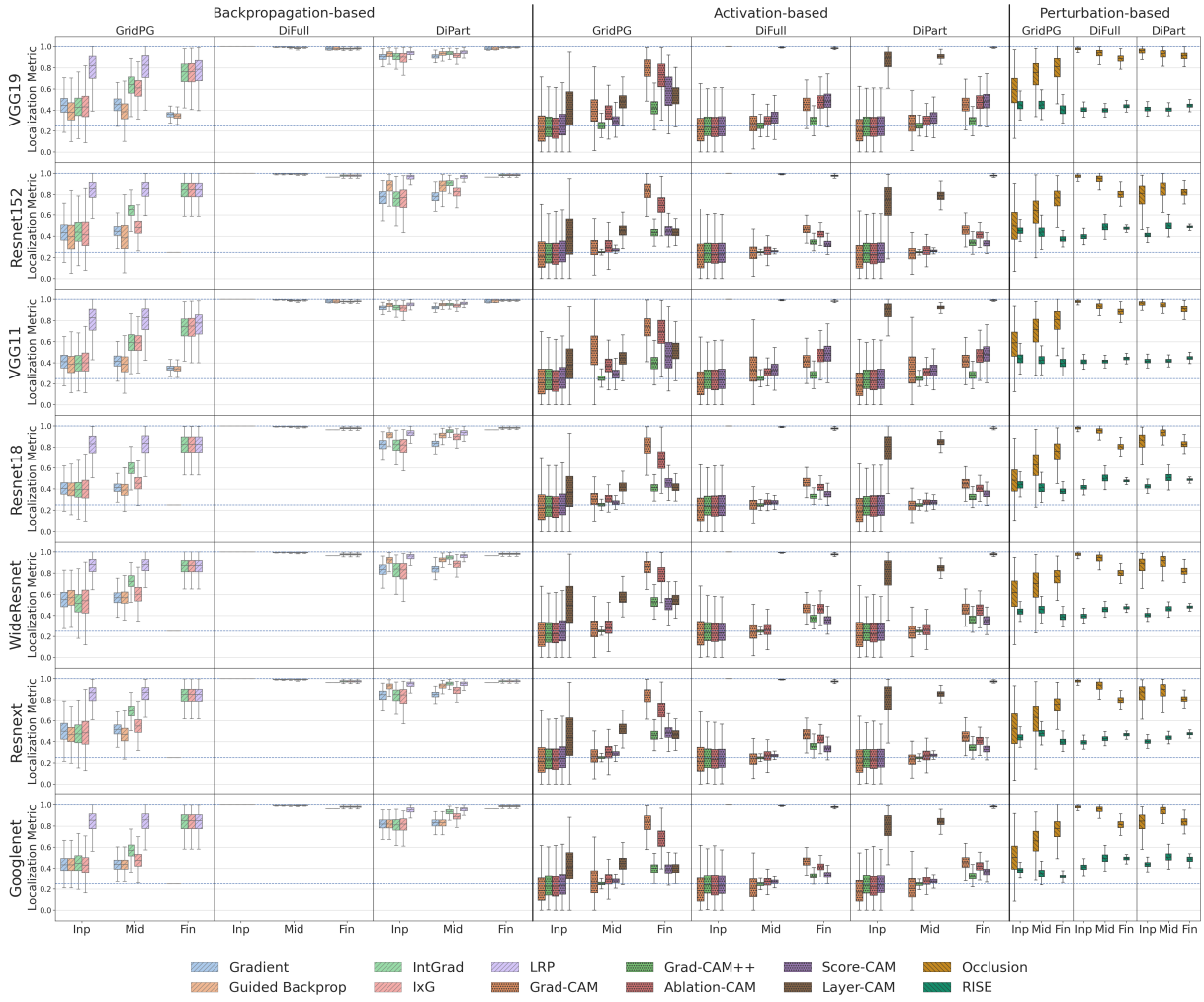


Figure A.1: **Quantitative Results on ImageNet**. We evaluate the localisation scores for each attribution method at the input (Inp), middle (Mid), and final (Fin) convolutional layers, on each of GridPG, DiFull, and DiPart using seven models: backpropagation-based (*left*), activation-based (*centre*), and perturbation-based (*right*) methods. The dashed horizontal lines mark localisation scores of 1.0 and 0.25, corresponding to perfect and random localisation.

The performance on DiFull and DiPart is very similar in all three settings and the three layers. The most significant difference between the two can be seen among the backpropagation-based methods and LayerCAM [JZH+21]. On DiFull, these methods show near-perfect localisation, since the gradients of the outputs from each classification head that are used to assign importance are zero with respect to weights and activations of all grid cells disconnected from that head. On the other hand, the receptive field of the convolutional layers can overlap adjacent grid cells in DiPart, and the gradients of the outputs from the classification heads can thus have non-zero values w.r.t. inputs and activations from these adjacent grid regions. This also results in decreasing localisation scores when moving backwards from the classifier.

Furthermore, the localisation scores for Gradient [SVZ14] and Guided Backpropagation [SDBR15] are constant at the final layer for most models, except VGG-19 and VGG-11. This is because in all these models, this layer is immediately followed by a global average pooling layer, due to which all activations at this layer get an equal share of the gradients.

A.2 QUALITATIVE RESULTS USING AGGATT

Here, we present additional qualitative results using our AggAtt evaluation and examples of attributions from each bin, for GridPG [BFS21] (Section A.2.1), DiFull (Section A.2.2), and DiPart (Section A.2.3).

Figure A.2: **Examples from each AggAtt bin for each method at the input layer on GridPG using VGG-19.** From each bin, the image and its attribution at the median position are shown.

A.2.1 GridPG

Figure A.2 and Figure A.3 show examples from the median position of each AggAtt bin for each attribution method at the input and final layers, respectively, evaluated on GridPG at the top-left grid cell using VGG-19 [SZ15]. At the input layer (Figure A.2), we observe that the **backpropagation-based methods** show noisy attributions that do not strongly localise to the top-left grid cell. This corroborates the poor quantitative performance of these methods at the input layer (Figure A.1, top). With the exception of LayerCAM [JZH+21], the **activation-based methods**, on the other hand, show strong attributions across all four grid cells, and localise very poorly. They appear to highlight the edges across the input irrespective of the class of each grid cell. This also agrees with the quantitative results (Figure A.1, middle), where the median localisation score of these methods is below the uniform attribution baseline. LayerCAM, being similar to IxG [SGK17], lies at the interface between activation and backpropagation-based methods, and also shows weak and noisy attributions. The **perturbation-based methods** visually show a high variance in attributions. While they localise well for about half the dataset (first three bins), the bottom half (last

three bins) shows noisy and poorly localised attributions, which again agrees with the quantitative results (Figure A.1, bottom). This further shows how evaluating on individual inputs can be misleading, and the utility of AggAtt for obtaining a holistic view across the dataset.

At the final layer (Figure A.3), attributions from Gradient [SVZ14] and Guided Backpropagation [SDBR15] are very noisy and only slightly concentrate at the top-left cell. The checkerboard-like pattern is a consequence of the max pooling operation after the final layer, which allocates all the gradient only to the maximum activation. Gradients from each position of the sliding classification kernel then get averaged to form the attributions. The localisation of IntGrad [STY17], IxG, GradCAM [SCD+17], and Occlusion [ZF14] improve considerably as compared to the input layer, which agrees with the quantitative results, and shows that diverse methods can show similar performance when compared fairly. The performance of the other activation-based methods and RISE [PDS18] improves to some extent, but is still poorly localised for around the half the dataset.

Figure A.3: **Examples from each AggAtt bin for each method at the final layer on GridPG using VGG-19.** From each bin, the image and its attribution at the median position are shown.

Finally, we show the AggAtt bins for all methods at all three layers using both VGG-19 and ResNet-152 [HZRS16] in Figure A.4. We see that the AggAtt bins reflect the trends observed in the examples in each bin, and serve as a useful tool for visualisation.

A.2.2 DiFull

Figure A.5 and Figure A.6 show examples from the median position of each AggAtt bin for each attribution method at the input and final layers, respectively, evaluated on DiFull at the top-left grid cell using VGG-19. At the input layer (Figure A.5), the backpropagation-based methods and LayerCAM show perfect localisation across the dataset. This is explained by the disconnected construction of DiFull, and agrees with the quantitative results shown in Figure A.1). The activation-based methods show very poor localisation that appear visually similar to the attributions observed on GridPG (Section A.2.1).

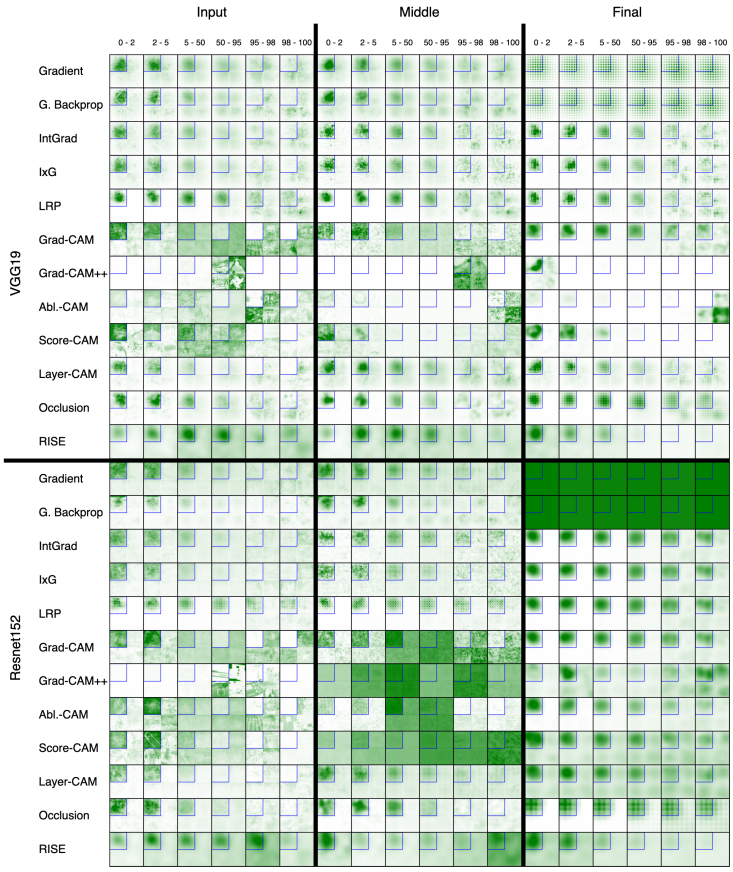


Figure A.4: **AggAtt Evaluation on GridPG** at the input, middle, and final layers using VGG-19 and ResNet-152.

Occlusion shows near-perfect localisation, since the placement of the classification kernel at any location not overlapping with the top-left grid cell does not influence the output in DiFull. RISE still produces noisy attributions across the dataset. While only the top-left grid cell influences the output, the use of random masks causes input regions that share masks with inputs in the top-left cell to also get attributed.

At the final layer (Figure A.6), the backpropagation-based methods and LayerCAM still show perfect localisation, for the same reason as discussed above. Attributions from Gradient and Guided Backpropagation show similar artifacts as seen with GridPG (Section A.2.1), but are localised to the top-left cell. The activation-based methods apart from LayerCAM concentrate their attributions at the top-left and bottom-right cells, particularly in the early bins. This is because both these cells contain images from the same class, and the weighing of activation maps by these methods causes both to be attributed, even though only the instance at the top-left influences the classification. Further, Occlusion and RISE show similar results as at the input layer. The attributions of Occlusion are noticeably lower in resolution, since the relative size of the occlusion kernel as compared to the activation map is much larger at the final layer.

Finally, we show the AggAtt bins for all methods at all three layers using both VGG-19 and ResNet-152 in Figure A.7, and see that they reflect the trends observed in the individual examples seen from each bin.

Figure A.5: **Examples from each AggAtt bin for each method at the input layer on DiFull using VGG-19.** From each bin, the image and its attribution at the median position are shown.

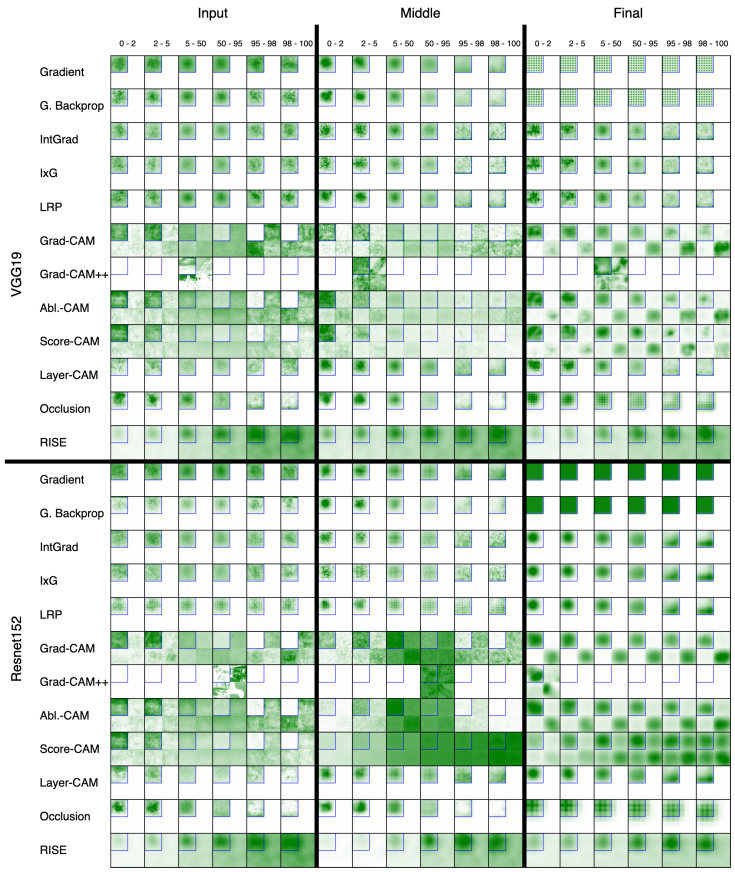


Figure A.7: **AggAtt Evaluation on DiFull** at the input, middle, and final layers using VGG-19 and ResNet-152.

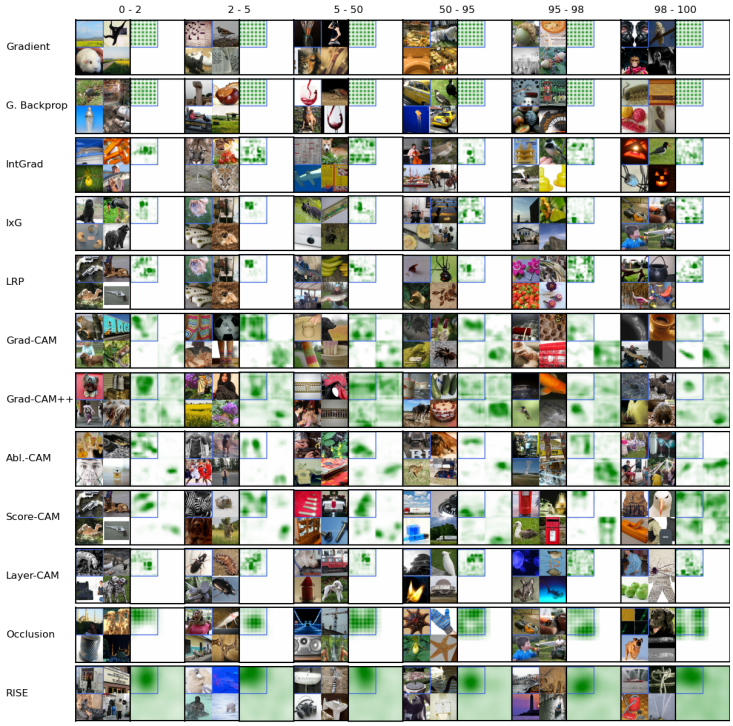


Figure A.6: **Examples from each AggAtt bin for each method at the final layer on DiFull using VGG-19.** From each bin, the image and its attribution at the median position are shown.

A.2.3 DiPart

Figure A.8 and Figure A.9 show examples from the median position of each AggAtt bin for each attribution method at the input and final layers, respectively, evaluated on DiPart at the top-left grid cell using VGG-19. In addition, Figure A.10 shows the AggAtt bins for all methods at all three layers using both VGG-19 and ResNet-152. As observed with the quantitative results (Section A.1, the performance seen visually on DiPart across the three layers is very similar to that on DiFull (Section A.2.2). However, they slightly differ in the case of the backpropagation-based methods and LayerCAM, particularly at the input layer (Figure A.8). This is because unlike in DiFull, the grid cells are only partially disconnected, and the receptive field of the convolutional layers can overlap adjacent grid cells to some extent. Nevertheless, as can be seen here, only a small boundary region around the top-left grid cell receives attributions, and the difference is not visually very perceivable. As such, DiPart setting can be thought of as a natural extension for DiFull, that mostly shares the requisite property without being an entirely constructed setting.

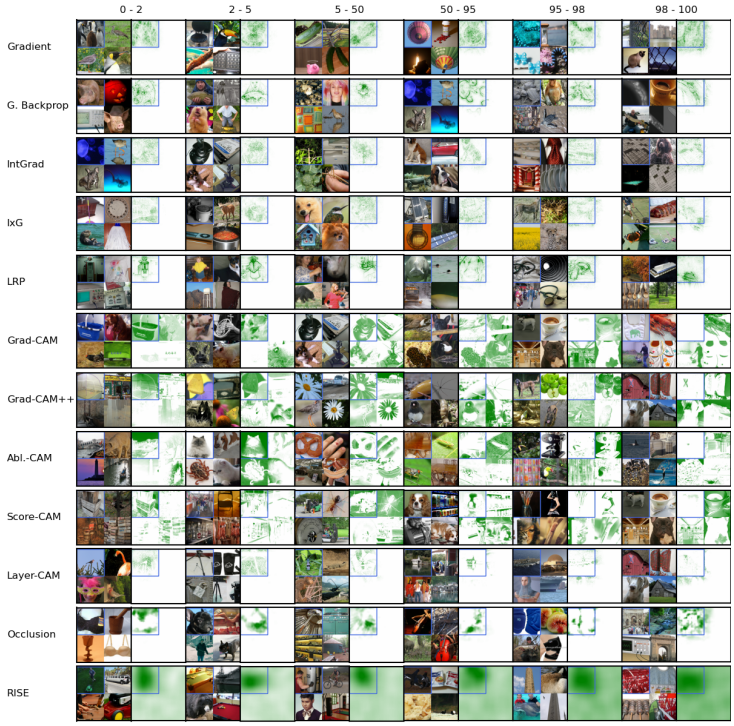


Figure A.8: **Examples from each AggAtt bin for each method at the input layer on DiPart using VGG-19.** From each bin, the image and its attribution at the median position are shown.

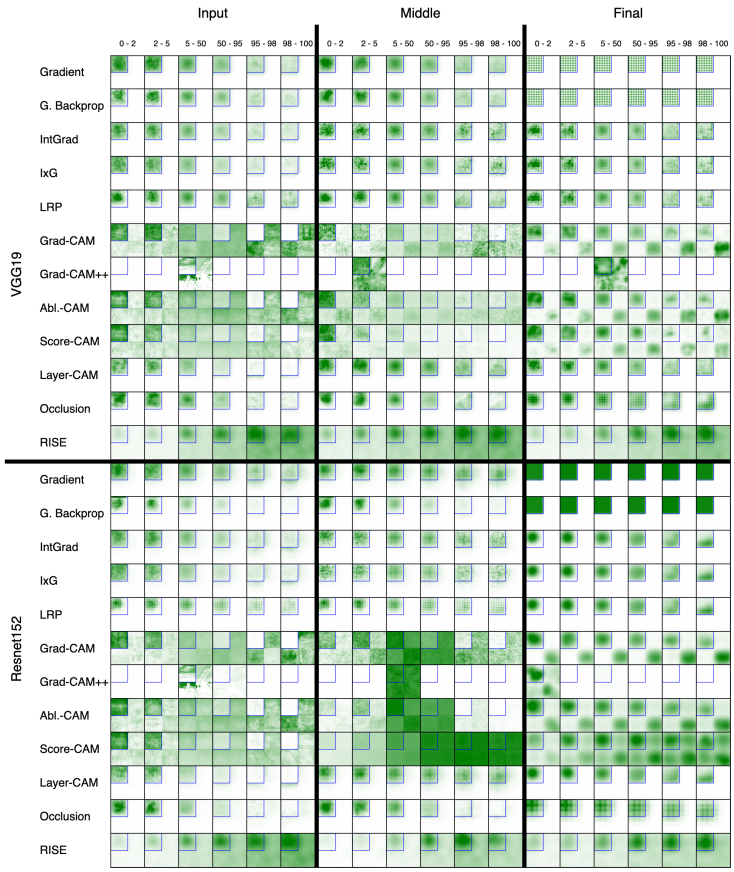


Figure A.10: **AggAtt Evaluation on DiPart** at the input, middle, and final layers using VGG-19 and ResNet-152.

Figure A.9: **Examples from each AggAtt bin for each method at the final layer on DiPart using VGG-19.** From each bin, the image and its attribution at the median position are shown.

A.3 CORRELATION BETWEEN ATTRIBUTIONS

From the quantitative (Figure A.1) and qualitative (Figure A.4) results, we observed that diverse methods perform similarly on GridPG [BFS21] both in terms of localisation score and through AggAtt visualisations when evaluated fairly. This was particularly the case with IntGrad [STY17], IxG [SGK17], GradCAM [SCD+17], and Occlusion [ZF14], when evaluated at the final layer. We also found (Section 4.3.3 in Chapter 4) that smoothing IntGrad and IxG (the result of which we call S-IntGrad and S-IxG) attributions evaluated at the *input layer* leads to visually and quantitatively similar performance as GradCAM evaluated at the *final layer*. In this section, we investigate this further, and study the correlation of these methods at the level of individual attributions. In particular, we compute the Spearman rank correlation coefficient of the localisation scores using VGG-19 [SZ15] of every pair of methods from each of the three layers. The results are shown in Figure A.11.

We observe that at the input layer (Figure A.11, top-left corner), the activation-based methods are poorly correlated with each other and with the backpropagation and perturbation-based methods. This also agrees with the poor localisation of these methods seen previously (Figure A.1, Figure A.2). The backpropagation-based and perturbation-based methods, on the other hand, show moderate to strong correlation amongst and with each other. Similar results can be seen when comparing methods at the middle layer with the input layer and the final layer (Figure A.11, edge centres). However, when compared at the middle layer (Figure A.11, middle), the activation-based methods still correlate poorly with other methods, but the strength of the correlation improves in general.

Further, when compared at the final layer (Figure A.11, bottom-right corner), all methods show moderate to strong correlations with each other. This could be because generating explanations at the final layer is a significantly easier task as compared to doing so at the input, since the activations are used as is and only

the classification layers' outputs are explained. The pairs with very strong positive correlation also show that attribution methods with diverse mechanisms can perform similarly when evaluated fairly. Finally, we observe that the activation-based methods at the final layer, instead of the input layer, correlate much better with the other methods at the input layer (Figure A.11, top-right, bottom-left).

	Unsmoothed	$K = 9$	$K = 17$	$K = 33$	$K = 65$	$K = 129$	$K = 257$
VGG-19	0.30	0.35	0.47	0.63	0.74	**0.78**	0.72
ResNet-152	0.16	0.19	0.22	0.32	0.46	**0.48**	0.44

Table A.1: **Spearman rank correlation coefficients between GradCAM at the final layer and S-IntGrad at the input layer on GridPG for varying degrees of smoothing.** We observe that the correlation improves significantly for both VGG-19 and ResNet-152 for larger kernels.

	Unsmoothed	$K = 9$	$K = 17$	$K = 33$	$K = 65$	$K = 129$	$K = 257$
VGG-19	0.20	0.20	0.26	0.36	**0.44**	0.41	0.31
ResNet-152	0.10	0.09	0.10	0.13	**0.18**	0.16	0.07

Table A.2: **Spearman rank correlation coefficients between GradCAM at the final layer and S-IxG at the input layer on GridPG for varying degrees of smoothing.** We observe that the correlation improves for VGG-19, but not much for ResNet-152.

We also observe that S-IntGrad and S-IxG at the input layer correlate well with the best-performing methods (IntGrad, IxG, GradCAM, Occlusion) at the final layer. Further, this marks a significant improvement when compared with IntGrad and IxG at the input layer. For example, IntGrad at the input layer compared with GradCAM at the final layer results in a correlation coefficient of 0.34, while S-IntGrad results in a correlation coefficient of 0.80.

We further study the effect of smoothing in Tables A.1 and A.2. We observe that the correlation between S-IntGrad and S-IxG improves significantly over IntGrad and IxG for VGG-19 when using large kernels. However, for ResNet-152 [HZRS16], the improvement for S-IxG is very small. This agrees with the quantitative localisation performance of these methods (Section 4.3.3 in Chapter 4). This shows that beyond aggregate visual similarity and quantitative performance, smoothing IntGrad and IxG can produce explanations at the input layer that are individually similar to GradCAM at the final layer, while also explaining the full network and performing significantly better on DiFull. We further visually compare the impact of smoothing in Section A.4.

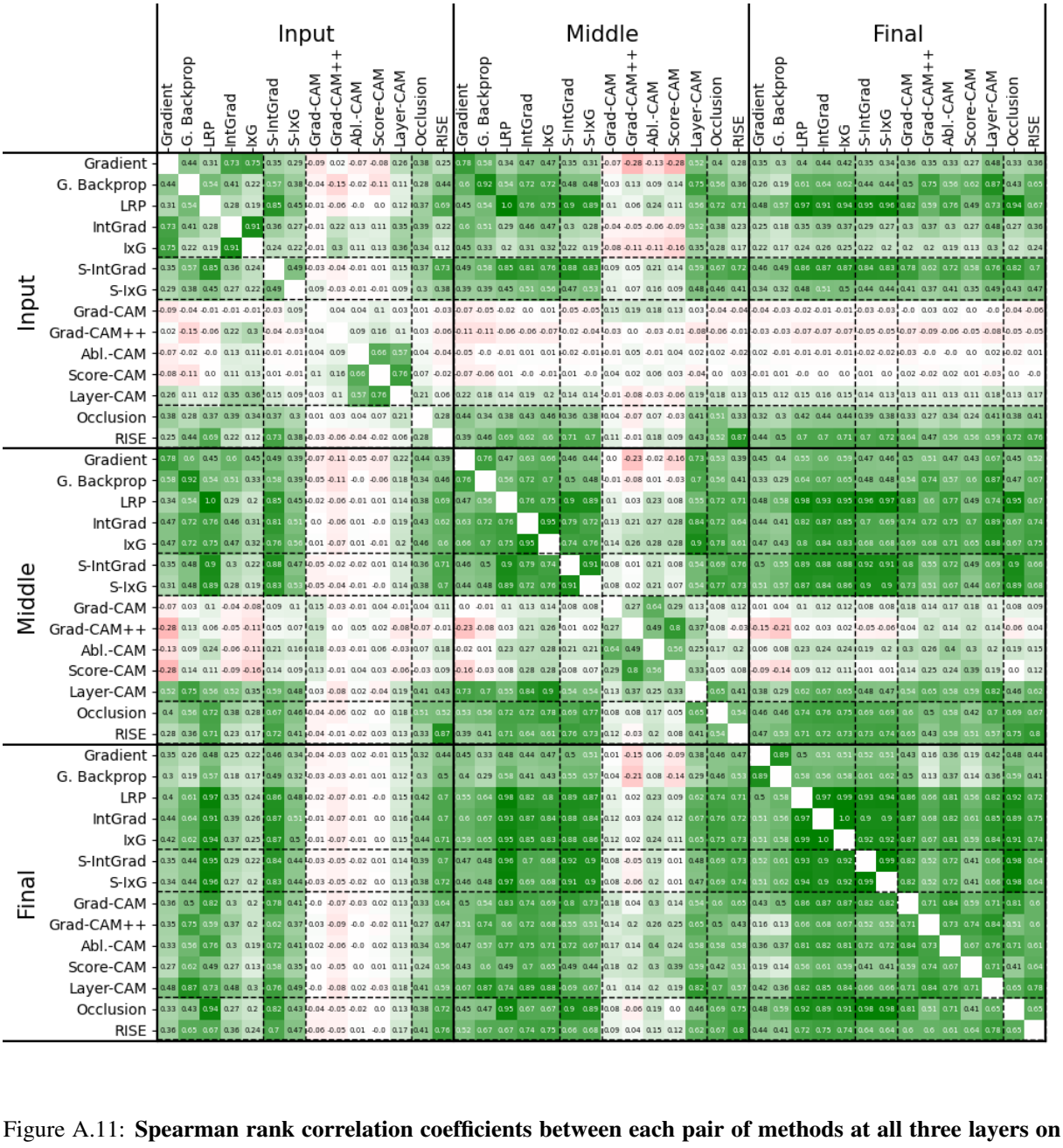


Figure A.11: **Spearman rank correlation coefficients between each pair of methods at all three layers on GridPG for VGG-19.** We observe that diverse methods (IntGrad, IxG, GradCAM, Occlusion) correlate strongly when evaluated at the final layer (bottom-right), which agrees with their similar quantitative and AggAtt performance. Further, S-IntGrad and S-IxG at the input layer correlate significantly stronger with GradCAM at the final layer as compared to IntGrad and IxG at the input layers. This, combined with the similarity in quantitative and AggAtt performance, shows the utility of smoothing in obtaining attributions that localise well while also performing much better in not attributing impossible regions specified by DiFull.

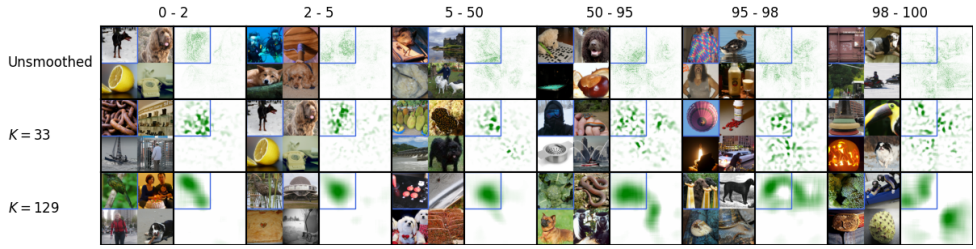
A.4 Impact of Smoothing Attributions

In this section, we explore the impact of smoothing attributions. First, we briefly discuss a possible reason for the improvement in localisation of attributions after smoothing (Section A.4.1). Then, we visualise the impact of smoothing through examples and AggAtt visualisations (Section A.4.2). Further, we also compare the performance of GradCAM [SCD+17] at the final layer with S-IntGrad and S-IxG at the input layer across the same examples from each bin and show their similarities across bins (Section A.4.3).

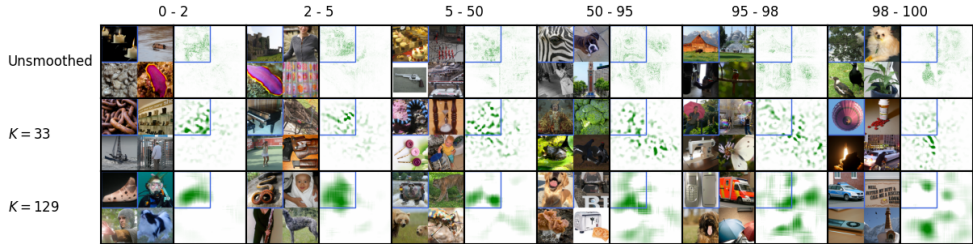
A.4.1 Effect of Smoothing

We believe that our smoothing results highlight an interesting aspect of piece-wise linear models (PLMs), which goes beyond mere practical improvements. For PLMs (such as the models used here), IxG [SGK17] yields the exact pixel contributions according to the linear mapping given by the PLM. In other words, the sum of IxG attributions over all pixels yields exactly (ignoring biases) the model output. If the effective receptive field of the model is small (cf. [LLUZ16]), sum pooling IxG with a kernel of the same size accurately computes the model's local output (apart from the influence of bias terms). Our method of smoothing IxG with a Gaussian kernel performs a weighted average pooling of attributions in the local region around each pixel, which produces a similar effect and appears to summarise the effect of the pixels in the local region to the model's output, which leads to less noisy attributions and better localisation.

A.4.2 AggAtt Evaluation after Smoothing



(a) S-IntGrad



(b) S-IxG

Figure A.12: **Examples from each AggAtt bin after smoothing IntGrad (a) and IxG (b) attributions on GridPG using VGG-19.** From each bin, the image and its attribution at the median position are shown.

In Section A.4.2 and Figure A.12, we show examples from each AggAtt bin for S-IntGrad and S-IxG at the input layer for two different kernel sizes, and compare with IntGrad [STY17] and IxG at the input layer respectively. We observe that the localisation performance significantly improves with increasing kernel size, and produces much stronger attributions for the target grid cell. In Figure A.13, we show the AggAtt bins for these methods on both VGG-19 [SZ15] and ResNet-152 [HZRS16]. We see that this reflects the trends seen from the examples, and also clearly shows the relative ineffectiveness of smoothing

IxG for ResNet-152 (Figure A.13 bottom right and Table A.2).

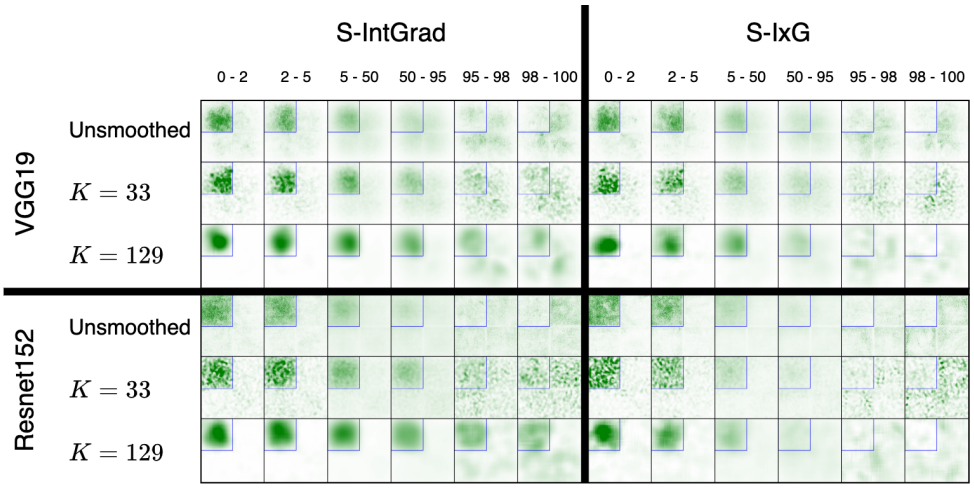
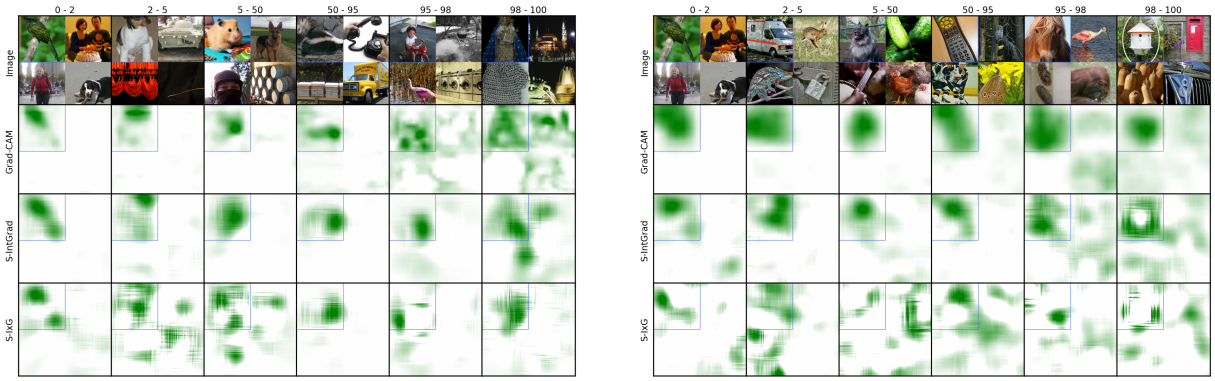


Figure A.13: **Impact of smoothing IntGrad and IxG visualised via AggAtt.** VGG-19 and ResNet-152 on GridPG.

A.4.3 Comparing GradCAM with S-IntGrad and S-IxG



(a) VGG19

(b) Resnet152

Figure A.14: **Comparing GradCAM and S-IntGrad/ S-IxG**. Example attributions from each AggAtt bin of GradCAM at the final layer compared with corresponding attributions from S-IntGrad and S-IxG at the input layer with $K = 129$, using VGG-19 and ResNet-152 on GridPG. We observe that S-IntGrad and S-IxG show visually similar examples to GradCAM across bins for VGG-19. While S-IntGrad also performs similarly for ResNet-152, S-IxG produces more noisy attributions.

We now compare GradCAM at the final layer with S-IntGrad and S-IxG at the input layer with $K = 129$ on the same set of examples (Figure A.14). We pick an example from each AggAtt bin of GradCAM, and evaluate all three methods on them. From Figure A.14, we observe that the three methods produce visually similar attributions across the AggAtt bins. While the attributions of S-IntGrad and S-IxG are somewhat coarser than GradCAM, particularly for the examples in the first few bins, they still concentrate around similar regions in the images. Interestingly, they perform similarly even for examples where GradCAM does not localise well, i.e., in the last two bins. Finally, we again see that S-IxG using ResNet-152 performs relatively worse as compared to the other methods (as also seen in Table A.2).

A.5 Quantitative Evaluation on All Layers

As discussed in Section 4.2, we evaluate trends in localisation performance across the full range of network depths across all seven models. Full results on all attribution methods can be found in Figure A.15.

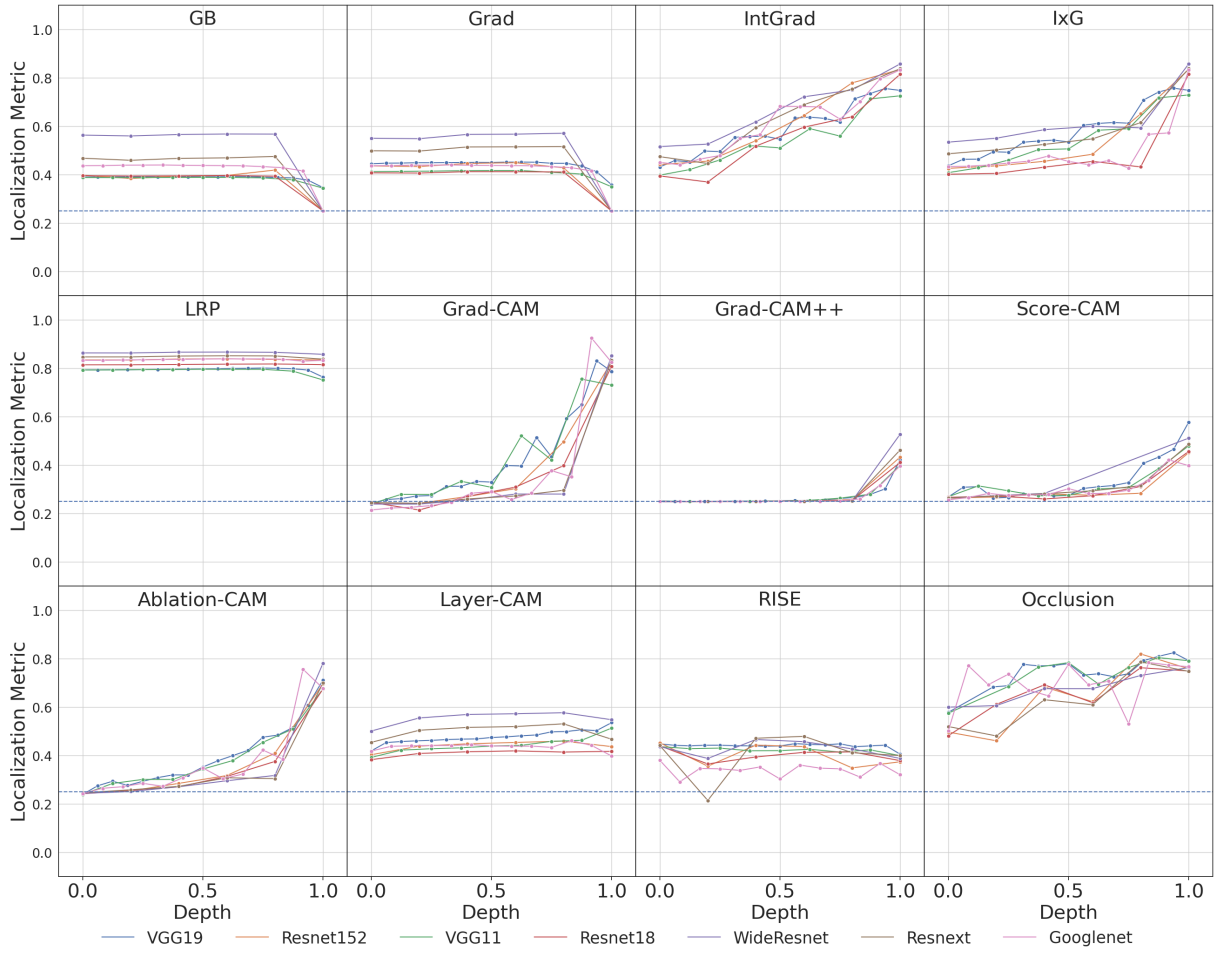


Figure A.15: **Mean Localisation performance layer-wise across seven models on all attribution methods**. For each method and network, we plot the mean localisation score at at several depths. The x-axis shows the fraction of the total network depth (0 - input, 1 - final layer).

Additionally, Figures A.16 and A.17 show the results on evaluating at each convolutional layer of VGG-11 [SZ15] and each layer block of ResNet-18 [HZRS16], visualised via boxplots to show the performance across the dataset. We find that the performance on the remaining layers is consistent with the trend observed from the three chosen layers in our experiments.

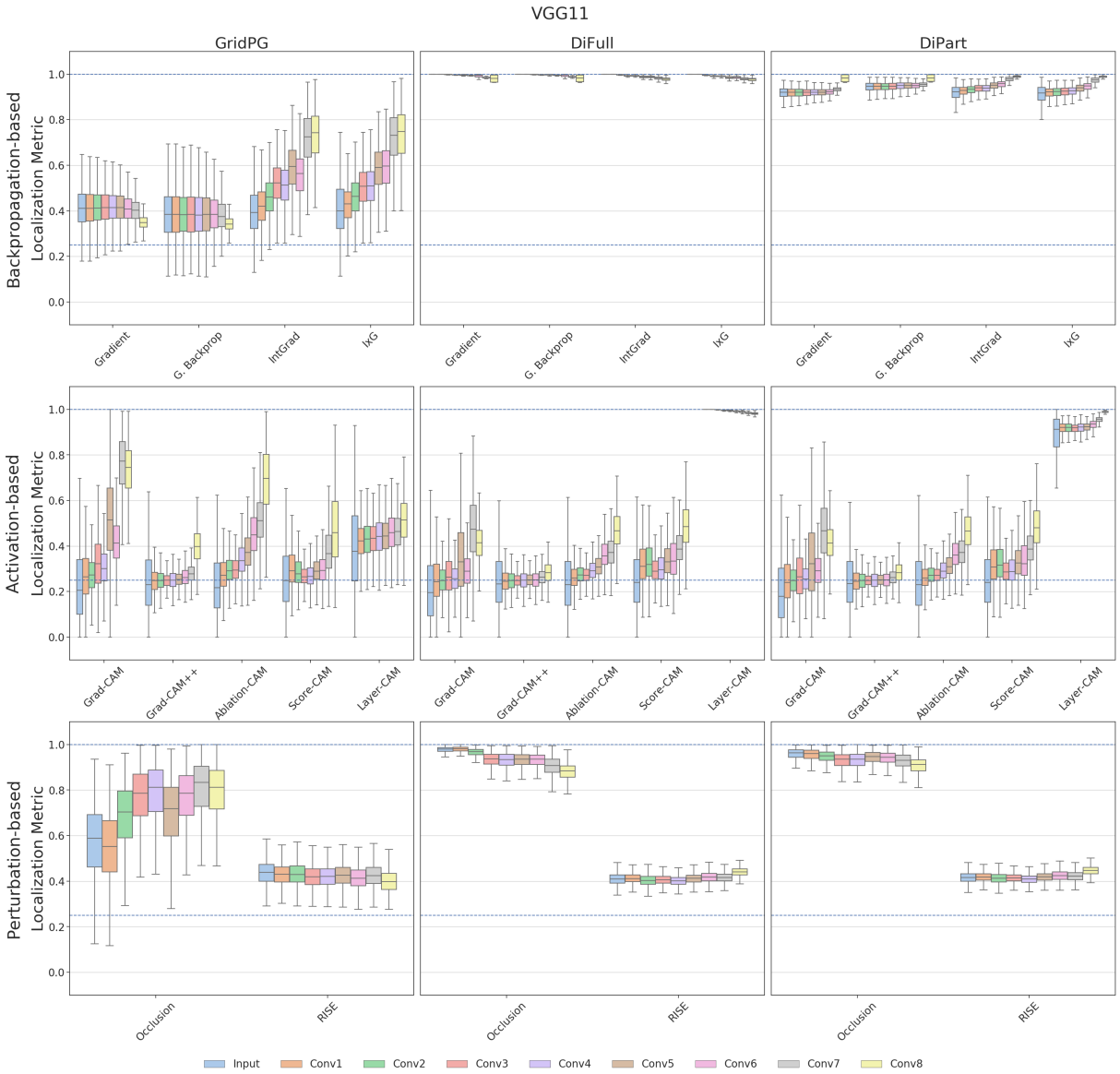


Figure A.16: **Quantitative Results for VGG-11 across all convolutional layers**. We evaluate the localisation scores each attribution method at the input and each convolutional layer of VGG-11, on each of GridPG, DiFull, and DiPart. *Top:* Backpropagation-based methods. *Middle:* Activation-based methods. *Bottom:* Perturbation-based methods. The two horizontal dotted lines mark localisation scores of 1.0 and 0.25, which correspond to perfect and random localisation, respectively. We find that the trends in performance corroborate with those seen across the selected input, middle, and final layers in our experiments.

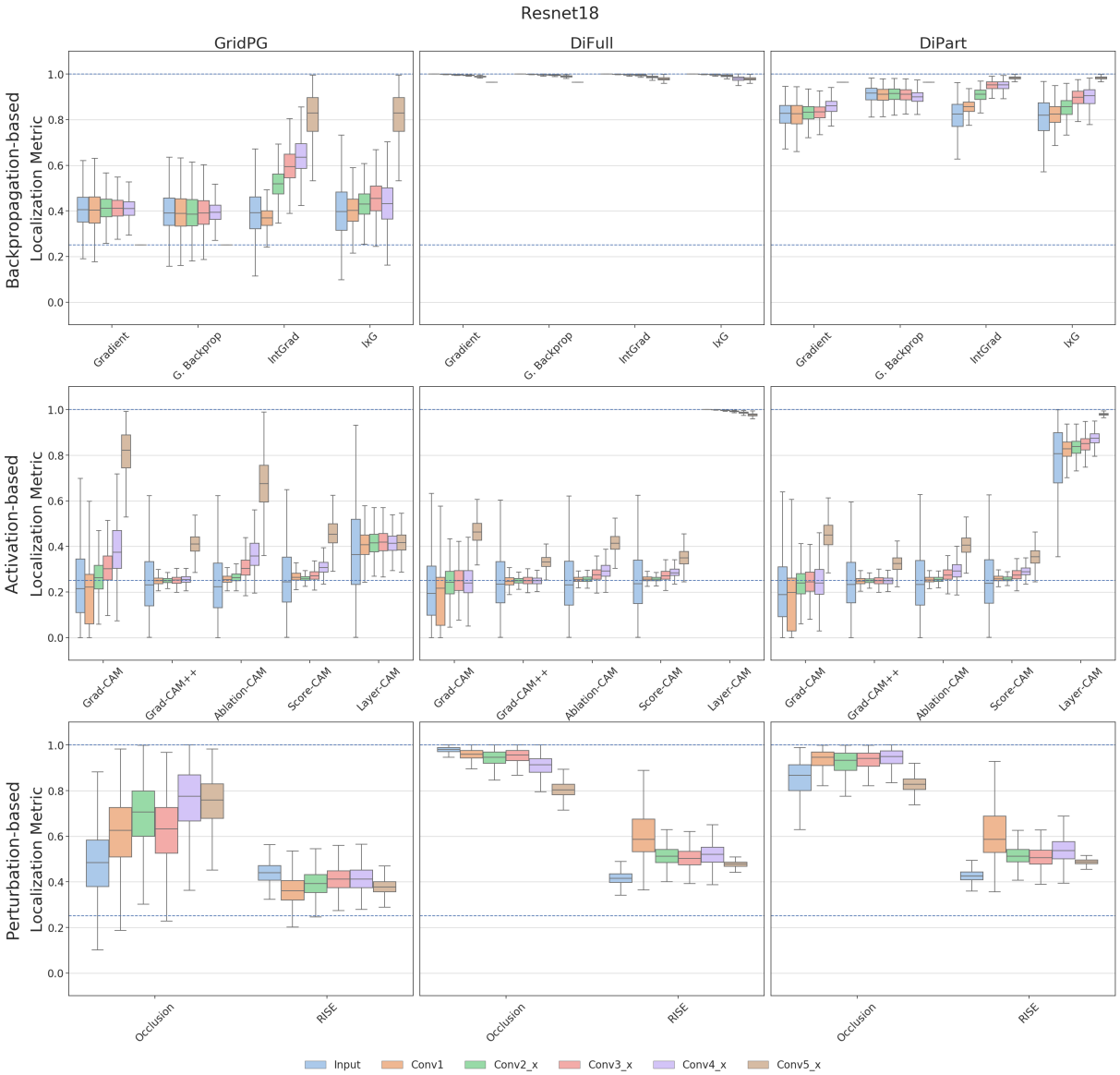


Figure A.17: **Quantitative Results for ResNet-18 across all convolutional layer blocks**. We evaluate the localisation scores each attribution method at the input and each convolutional layer of ResNet-18, on each of GridPG, DiFull, and DiPart. *Top:* Backpropagation-based methods. *Middle:* Activation-based methods. *Bottom:* Perturbation-based methods. The two horizontal dotted lines mark localisation scores of 1.0 and 0.25, which correspond to perfect and random localisation, respectively. We find that the trends in performance corroborate with those seen across the selected input, middle, and final layers in our experiments.

A.6 Computational Cost

Unlike GridPG [BFS21], the DiFull setting involves passing each grid cell separately through the network. In this section, we compare the computational costs of GridPG, DiFull, and DiPart, and show that it is similar across the three settings. Let the input be in the form of a $n \times n$ grid. Each setting consists of a CNN module, which obtains features from the input, and a classifier module, which provides logits for each cell in the grid using the obtained features. We analyse each of these modules one by one.

CNN Module: In GridPG and DiPart, the entire grid is passed through the CNN module as a single input. On the other hand, in DiFull, each grid cell is passed separately. This can be alternatively viewed as stacking each of the n^2 grid cells along the batch dimension before passing them through the network. Consequently, the inputs in the DiFull setting have their widths and heights scaled by a factor of $\frac{1}{n}$, and the batch size scaled by a factor of n^2. Since the operations within the CNN module scale linearly with input size, the computational cost for each grid cell in DiFull is $\frac{1}{n^2}$ times the cost for the full grid in GridPG and DiPart. Since there are n^2 such grid cells, the total computational cost for the CNN module of DiFull equals that of GridPG and DiPart.

Classifier Module: The classifier module in the DiFull and DiPart settings consists of n^2 classification heads, each of which receives features corresponding to a single grid cell. On the other hand, the GridPG setting uses a classifier kernel over the composite feature map for the full grid. Let the dimensions of the feature map for a single grid cell be $d \times d$. This implies that in GridPG, using a stride of 1, the classification kernel slides over $((n-1)d+1)^2$ windows of the input, each of which results in a call to the classifier module. In contrast, in DiFull and DiPart, the classifier module is called only n^2 times, one for each head. This shows that the computational cost of DiFull and DiPart for the classifier module and the pipeline as a whole is at most as much as of GridPG.

A.7 COMPARISON WITH SMOOTHGRAD

We find that smoothing IntGrad [STY17] and IxG [SGK17] attributions with a Gaussian kernel can lead to significantly improved localisation, particularly for networks without batch normalisation layers [IS15]. As discussed in Section A.4.1, we believe this to be because smoothing summarises the effect of inputs in a local window around each pixel to the output logit, and reduces noisiness of attributions. Prior approaches to address noise in attributions include SmoothGrad [STK+17], which involves adding Gaussian noise to an input and averaging over attributions from several samples. Here, we compare our smoothing with that of SmoothGrad. Figure A.18 shows that our methods (S-IntGrad, S-IxG) show significantly better GridPG [BFS21] localisation than SmoothGrad on IntGrad and IxG, except in the case of IxG with ResNet-18 [HZRS16], where our smoothing does not improve localisation likely due to the presence of batch normalisation layers. The scores on DiFull decrease to an extent since our Gaussian smoothing allows attributions to "leak" to neighbouring grid cells. These results are corroborated by AggAtt visualisations in Figure A.19. We also note that SmoothGrad requires significantly higher computational cost than our approach, as attributions need to be generated for several noisy samples of each input, and is also sensitive to the choice of hyperparameters such as the noise percentage and the number of samples.

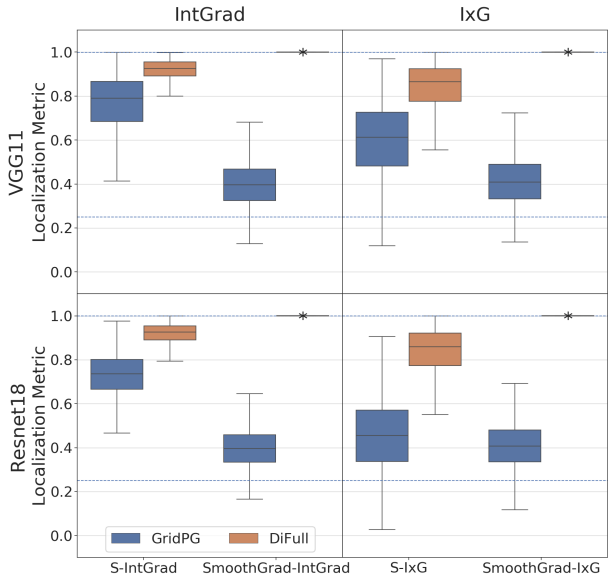


Figure A.18: **Quantitative Results comparing our smoothing with SmoothGrad using VGG-11**. We evaluate each attribution method at the input layer. For S-IntGrad and S-IxG, we use $K = 129$. For SmoothGrad, we use the best performing configuration after varying the noise percentage from 1% to 30%, and use 15 samples per input. *Top:* Results on VGG-11. *Bottom:* Results on ResNet-18. We use the "*" symbol to show boxes that collapse to a single point, for better readability.

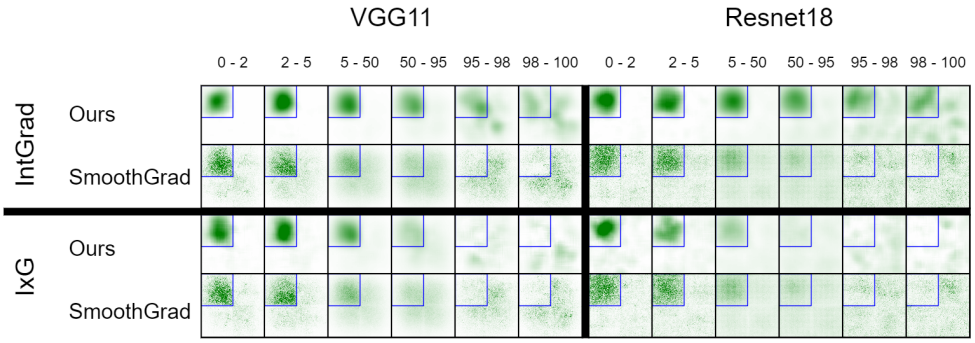


Figure A.19: **Comparing SmoothGrad with S-IntGrad and S-IxG.** AggAtt visualisations of our smoothing (S-IntGrad, S-IxG) compared with SmoothGrad applied on IntGrad and IxG using VGG-11 and ResNet-18 on GridPG.

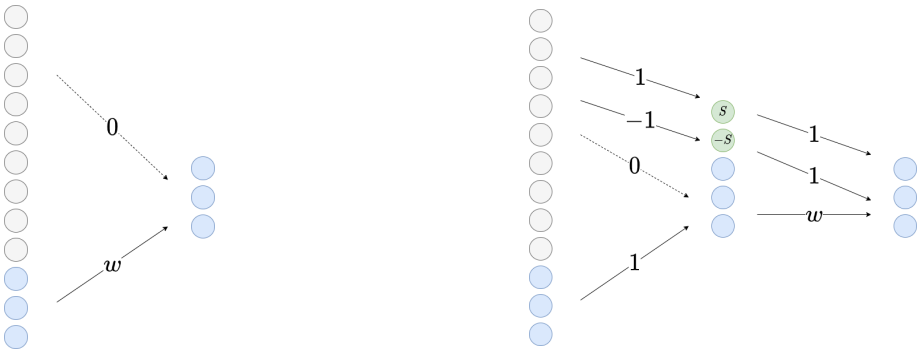
A.8 LRP: LACK OF IMPLEMENTATION INVARIANCE

In this section, we provide an illustration on the impact of the lack of implementation invariance [STY17] in the z^+ propagation rule of LRP. Using our DiFull setting, we provide a general construction of a transform that results in a functionally equivalent model but causes LRP with the z^+ rule to shift nearly all attributions to the disconnected regions.

The transform is illustrated in Figure A.20. In the DiFull setting, each classification head is connected to its corresponding grid cell's feature map with the weights from the pre-trained model, and is disconnected (alternatively, connected with zero weights) to all other grid cells (Figure A.20a). For the transform, we introduce an additional linear layer $l_{c'}$ between the final feature activations f and the first classification layer l_c, as shown in Figure A.20b.

Let the number of neurons in l_c be k. We construct $l_{c'}$ with $k+2$ neurons, as shown. We use k neurons to map the original linear transformation back to l_c, by using an identity transform between the connected regions in f and $l_{c'}$, and using zero weights for the disconnected regions. The two additional neurons are used to aggregate the activations from all the disconnected regions of f, one each with positive and negative sign. These neurons are then connected with weight 1 to l_c. As a result, the contributions from the disconnected regions cancel out in the output of l_c, and the model remains functionally equivalent.

However, when applying LRP with the z^+ propagation rule, relevance is distributed only among neurons that contribute positively to the output. Consequently, when propagating from $l_{c'}$ to l_c, most of the relevance is transferred to the new neuron that aggregates positive attributions from the disconnected regions. Since the regions are fully disconnected for all further propagation, this causes most attributions to now be given to the disconnected regions, as shown in Figure A.21.



(a) Original (b) Transformed

Figure A.20: **Transform that inserts a new layer between the final feature layer and the first classification layer in the DiFull setting.** The resultant model is functionally equivalent to the original model. We add a new layer with a neuron that each aggregate all activations from all regions with zero connections with positive and negative sign respectively. The contributions from these two neurons cancel out in the next layer, keeping the model functionally equivalent. However, the z^+ rule only considers neurons that contribute positively to the output, causing almost all relevance to flow through the new positive neuron and to the disconnected regions (as shown in Figure A.21).

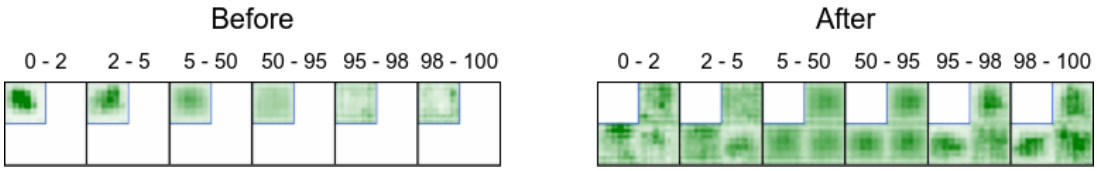


Figure A.21: **LRP Attributions for VGG-19 in the DiFull setting before and after the transform.** As the z^+ propagation rule is not implementation invariant, appropriately transforming the model to a functionally equivalent model can result in vastly different attribution maps.

A.9 Implementation Details

A.9.1 Dataset

As described in Section 4.2 in Chapter 4, we obtain 2,000 attributions for each attribution method on each of GridPG [BFS21], DiFull, and DiPart, using inputs consisting of four subimages arranged in 2×2 grids. For GridPG, since we evaluate on all four subimages, we do this by constructing 500 grid images after randomly sampling 2,000 images from the validation set. Each grid image contains subimages from four distinct classes. On the other hand, for DiFull and DiPart, we place images of the same class at the top-left and bottom-right corners to test whether an attribution method simply highlights class-related features, irrespective of them being used by the model. Therefore, we evaluate only on these two grid locations. In order to obtain 2,000 attributions as with GridPG, for these two settings, we construct 1,000 grid images by randomly sampling 4,000 images from the validation set.

A.9.2 Models and Attribution Methods

We implement our settings using PyTorch [PGM+19], and use pretrained models from Torchvision [PGM+19]. We use implementations from the Captum library [KMM+20] for Gradient [SVZ14], Guided Backpropagation [SDBR15], IntGrad [STY17], and IxG [SGK17], from the Zennit library [ANS+21] for LRP [BBM+15], and from [BFS21] for Occlusion [ZF14] and RISE [PDS18]. For Gradient and Guided Backpropagation, the absolute value of the attributions are used. All attributions across methods are summed along the channel dimensions before evaluation.

Occlusion involves sliding an occlusion kernel of size K with stride s over the image. As the spatial dimensions of the feature maps decreases from the input to the final layer, we select different values of K and s for each layer. In our experiments, we use $K = 16, s = 8$ for the input, and $K = 5, s = 2$ for the middle and final layers.

RISE generates attributions by occluding the image using several randomly generated masks and weighing them based on the change in the output class confidence. In our experiments, we use $M = 1000$ masks. We use fewer masks than [PDS18] to offset the increased computational cost from using 448×448 images, but found similar results from a subset of experiments with $M = 6000$.

The γ-rule for LRP-Composite as originally defined expects positive inputs. So, to enable its use with negative inputs that may result from using ResNet model, we use the generalised version of the γ-rule (proposed in [AKW+23]) that is implemented in the Zennit library [ANS+21].

A.9.3 Localisation Metric

In our quantitative evaluation, we use the same formulation for the localisation score as proposed in GridPG (see Section 4.1.1, Equation (4.1)). Let $A^+(p)$ refer to the positive attribution given to the p^{th} pixel. The localisation score for the subimage x_i is given by:

$$L_i = \frac{\sum_{p \in x_i} A^+(p)}{\sum_{j=1}^{n^2} \sum_{p \in x_j} A^+(p)} \tag{A.1}$$

However, L_i is undefined when the denominator in Equation (A.1) is zero, i.e., $\sum_{j=1}^{n^2} \sum_{p \in x_j} A^+(p) = 0$. This can happen, for instance, when all attributions for an input are negative. To handle such cases, we set $L_i = 0$ in our evaluation whenever the denominator is zero.

A.9.4 AggAtt Visualisations

To generate our AggAtt visualisations, we sort attribution maps in the descending order of the localisation score and bin them into percentile ranges to obtain aggregate attribution maps (Section 4.1.2). However, we observe that when evaluating on DiFull, the backpropagation-based attribution methods show perfect localisation (Section 4.1.1), and all attributions share the same localisation score. In this scenario, and in all other instances when two attributions have the same localisation score, we break the tie by favouring maps that have stronger attributions in the target grid cell. We do this by ordering attributions with the same localisation score in the descending order of the sum of attributions within the target grid cell, i.e., the numerator in Equation (A.1).

Further, when producing the aggregate maps, we normalise the aggregate attributions using a common normalising factor for each method. This is done to accurately reflect the strength of the average attributions across bins for a particular method.

A.10 EVALUATION ON CIFAR10

In addition to ImageNet [DDS+09], we also evaluate using our settings on CIFAR10 [Kri09]. In this section, we present these results, and find similar trends in performance as on ImageNet. We first describe the experimental setup (Section A.10.1) used, and then show the quantitative results on GridPG [BFS21], DiFull, and DiPart (Section A.10.2) and some qualitative results using AggAtt (Section A.10.3).

A.10.1 Experimental Setup

Network Architecture. We use a modified version of the VGG-11 [SZ15] architecture, with the last two convolutional layers removed. Since the CIFAR10 inputs have smaller dimensions (32×32) than ImageNet (224×224), using all the convolutional layers results in activations with very small spatial dimensions, which makes it difficult to apply attribution methods at the final layer. After removing the last two convolutional layers, we obtain activations at the new final layer with dimensions 4×4 before pooling. We then perform our evaluation at the input (Inp), middle layer (Conv3) and the final layer (Conv6).

Data. We construct grid datasets consisting of 2×2 and 3×3 grids using images from the validation set classified correctly by the network with a confidence of at least 0.99. We obtain 4,000 (resp. 4,500) attributions for each method from the 2×2 (resp. 3×3) grid datasets respectively. As with ImageNet

(Section A.9.1), we evaluate on all grid cells for GridPG and only at the top-left and bottom-right corners on DiFull and DiPart. To obtain an equivalent 4,000 (resp. 4,500) attributions from using just the corners on DiFull and DiPart, we randomly sample 8,000 (resp. 20,250) images for the 2×2 (3×3) grid datasets, and construct 2,000 (2,250) composite images. Note that the CIFAR10 validation set only has a total of 10,000 images. Since we only evaluate at the two corners, we allow subimages at other grid cells to repeat across multiple composite images. However, no two subimages are identical within the same composite image.

A.10.2 Quantitative Evaluation on GridPG, DiFull, and DiPart

The results of the quantitative evaluation can be found in Figure A.22 for both 2×2 grids (left) and 3×3 grids (right). We observe that all methods perform similarly as on ImageNet (Figure A.1). Since localising on 3×3 grids poses a more challenging task, we observe generally poorer performance across all methods on that setting.

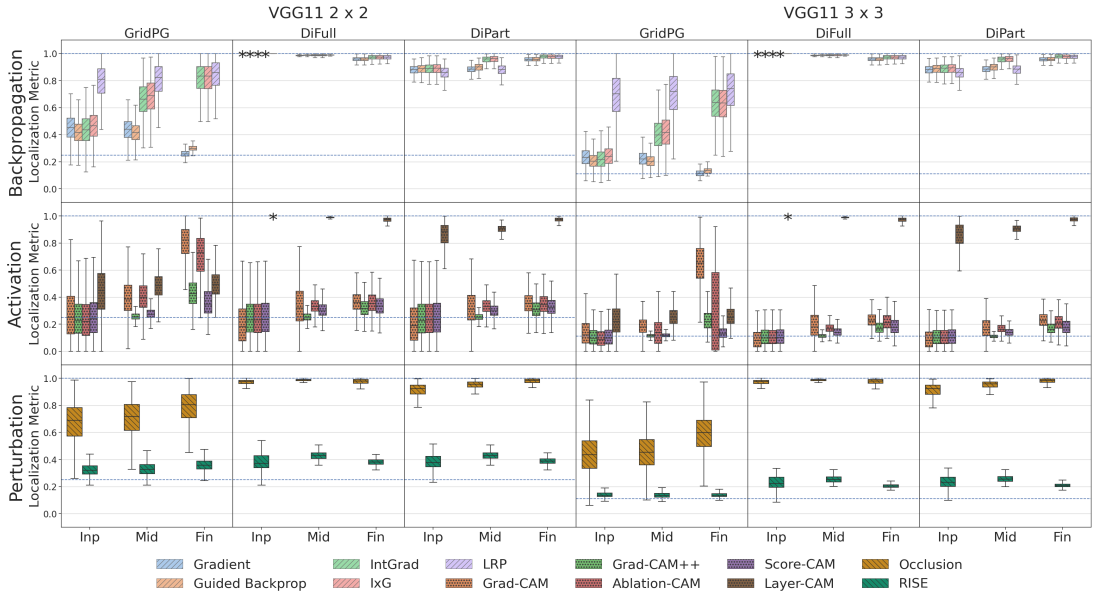


Figure A.22: **Quantitative Results on CIFAR10 using VGG-11**. We evaluate each attribution method at the input (Inp), middle (Mid), and final (Fin) convolutional layers, on each of GridPG, DiFull, and DiPart using 2×2 (left) and 3×3 (right) grids. *Top:* Results on backpropagation-based methods. *Middle:* Results on activation-based methods. *Bottom:* Results on perturbation-based methods. The two horizontal dotted lines mark localisation scores that correspond to perfect and random localisation, respectively, which equal scores of 0.25 and 0.11 respectively for 2×2 and 3×3 grids. We use the "*" symbol to show boxes that collapse to a single point, for better readability.

A.10.3 Qualitative Results Using AggAtt

In Figure A.23, we show AggAtt evaluations on 3×3 grids for a method each from the set of backpropagation-based (IxG [SGK17]), activation-based (GradCAM [SCD+17]), and perturbation-based (Occlusion [ZF14]) methods. Further, we show examples of attributions at the input and final layer on GridPG for these methods (Figures A.24 and A.25). We see that these show similar trends in their performance as on ImageNet (Section A.2).

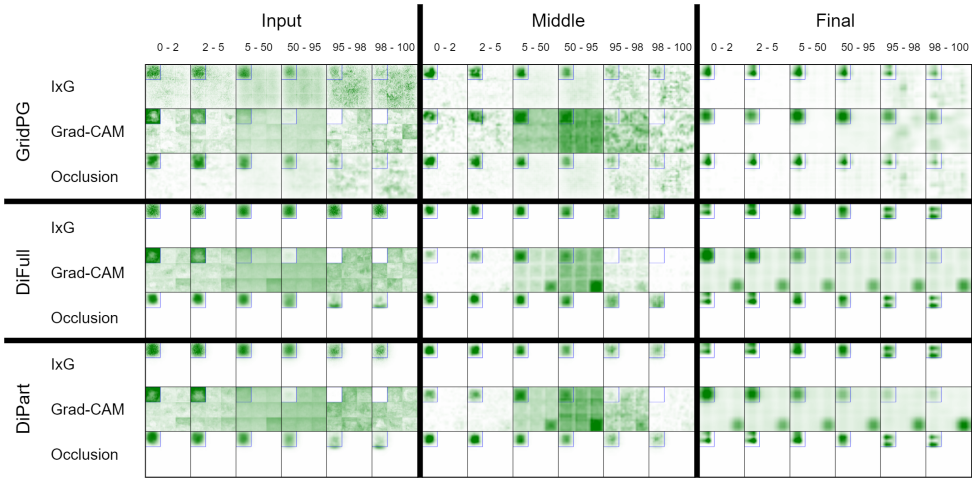


Figure A.23: **AggAtt Evaluation on GridPG for methods at the input, middle, and final layers using VGG-11 with the 3×3 grid dataset.**

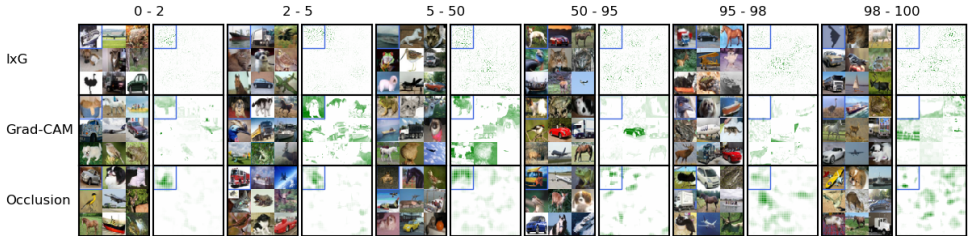


Figure A.24: **Examples from each AggAtt bin for each method at the input layer on GridPG using VGG-11 on 3×3 grids.** From each bin, the image and its attribution at the median position are shown.

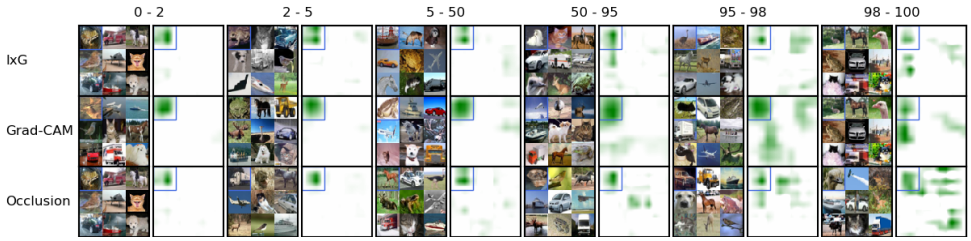


Figure A.25: **Examples from each AggAtt bin for each method at the final layer on GridPG using VGG-11 on 3×3 grids.** From each bin, the image and its attribution at the median position are shown.

APPENDIX — CODA NETWORKS

B

In this supplement to our work on Convolutional Dynamic Alignment Networks (CoDA Nets), we provide:

B.1 ADDITIONAL QUALITATIVE RESULTS

Additional ImageNet Examples.

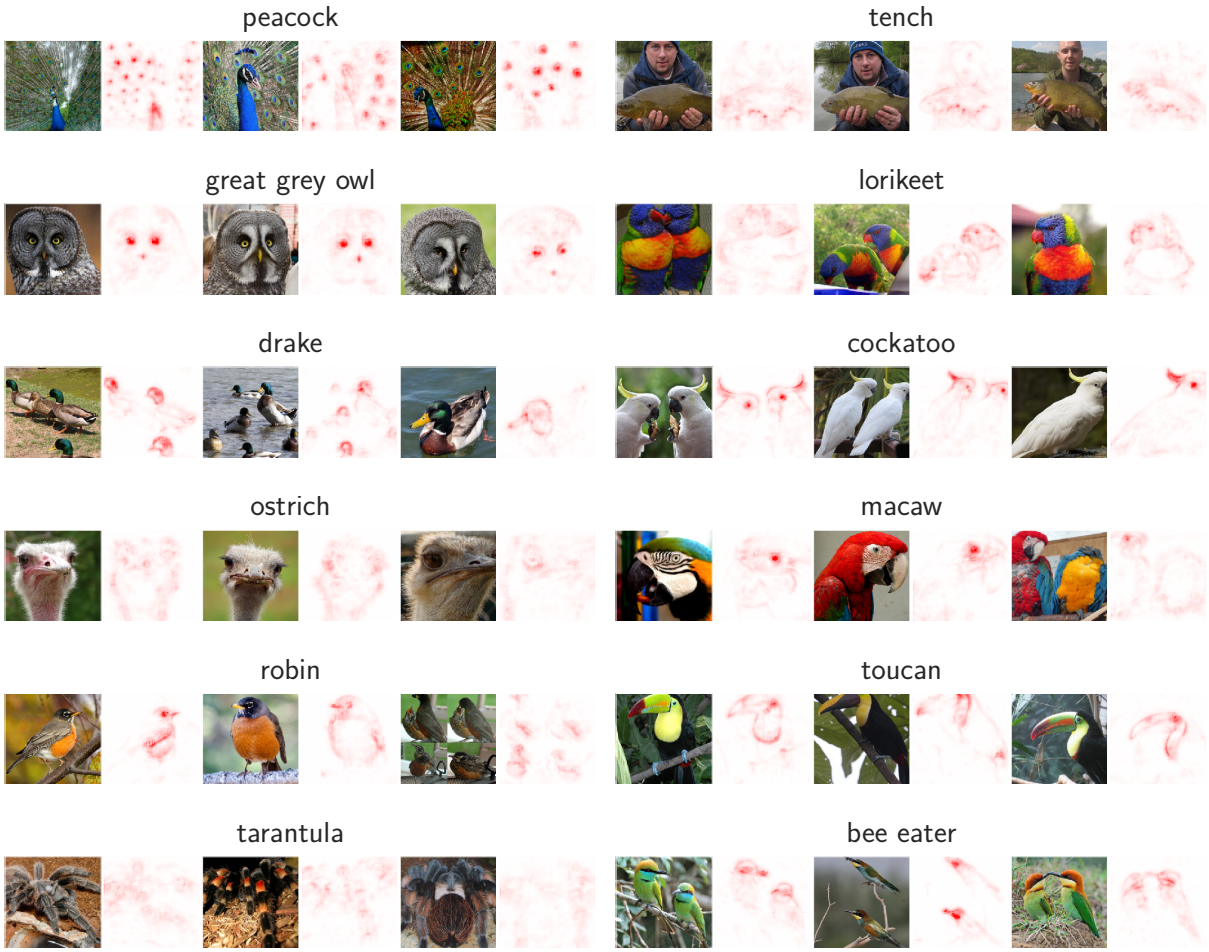


Figure B.1: Explanations for the three most confident examples of the first 12 classes when sorting the classes by the mean logits across the three most confidently classified samples per class. Positive (negative) contributions for the ground truth class are shown in red (blue).

In Figures B.1 and B.2 we present additional qualitative examples of the model-inherent contribution maps. In particular, we show the decomposition of the model predictions into input contributions for 32 out of 100 classes; for each class, we show the three most confidently classified images and show the classes in sorted order (by mean logit score across those three images). As can be seen, the explanations are not only very detailed, highlighting very fine-grained features such as the eyes in the feathers of the peacock (first row, Figure B.1) or the eyes of the great grey owl (second row, Figure B.1), but are also highly consistent across instances of the same class: e.g., from the contribution maps it becomes clear that the eyes of the great grey owl constitute the most discriminative feature for the model. These discriminative features vary significantly per class and are semantically meaningful: e.g., the model focuses on the yellow head feathers of the cockatoo or the green head of the drake (third row, Figure B.1), or the highly recognisable patterns of the feathers of the jay (second row, Figure B.2). As such, the model seems to rely on features that are consistent with the patterns a human would rely on, making the explanations easily interpretable.

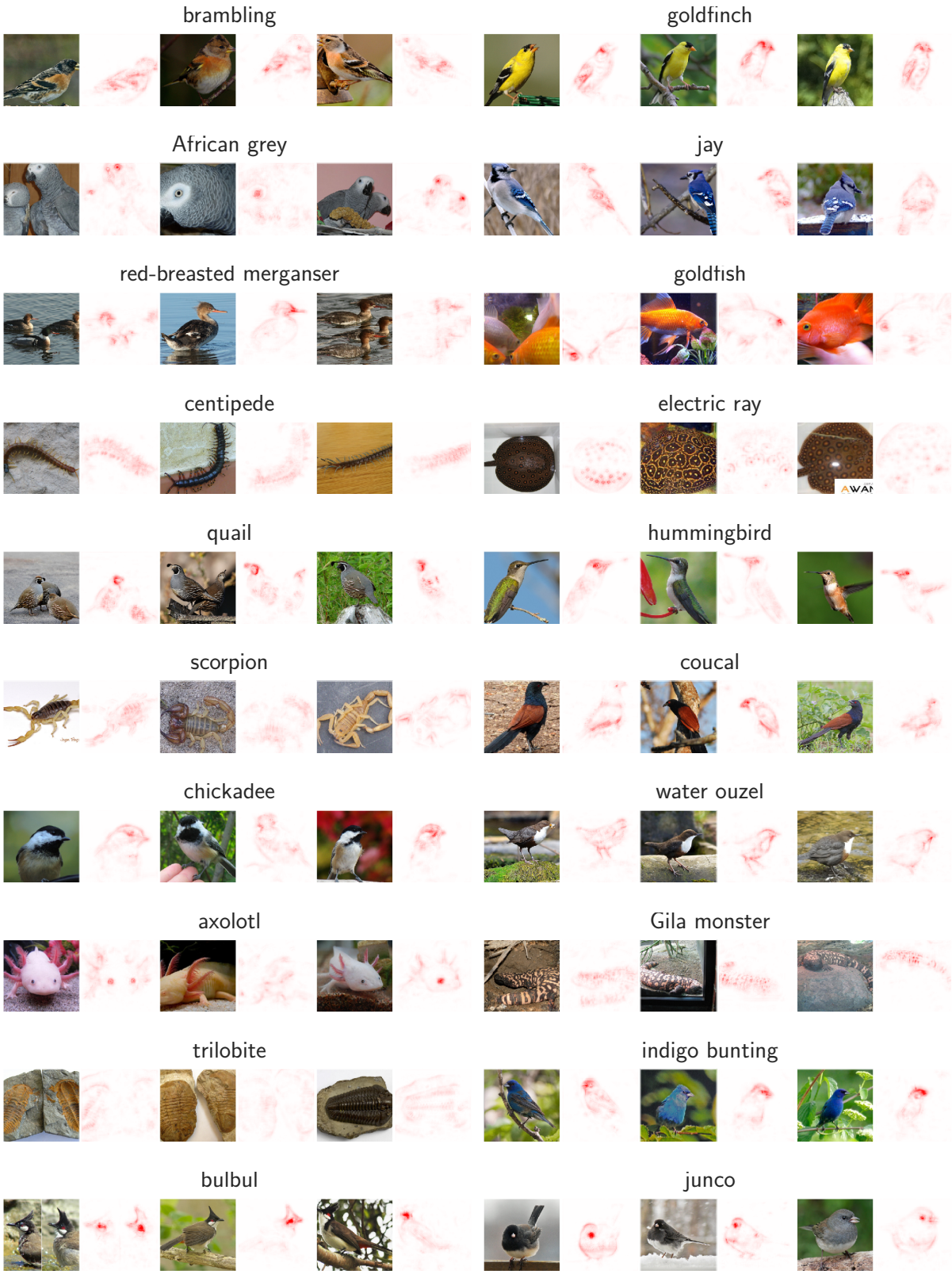


Figure B.2: Explanations for the three most confident examples of classes 16 to 32 when sorting the classes by the mean logits across the three most confidently classified samples per class. Positive (negative) contributions for the ground truth class are shown in red (blue).

In Figure B.3, we moreover show the contributions maps for the first 8 of the overall most confidently classified images next to the attribution maps from the post-hoc importance attribution methods for qualitative comparison. We note that GradCAM consistently highlights very similar regions to the CoDA Net contribution maps, but does so at a lower resolution. All contribution maps based on the CoDA Net use the same linear colour scale, which has been set to $(-v, v)$ with v the 99.99th percentile over all absolute values in the contributions maps for the three most confident predictions of each class. For reproducing the presented contribution maps and more, please visit github.com/moboehle/CoDA-Nets.

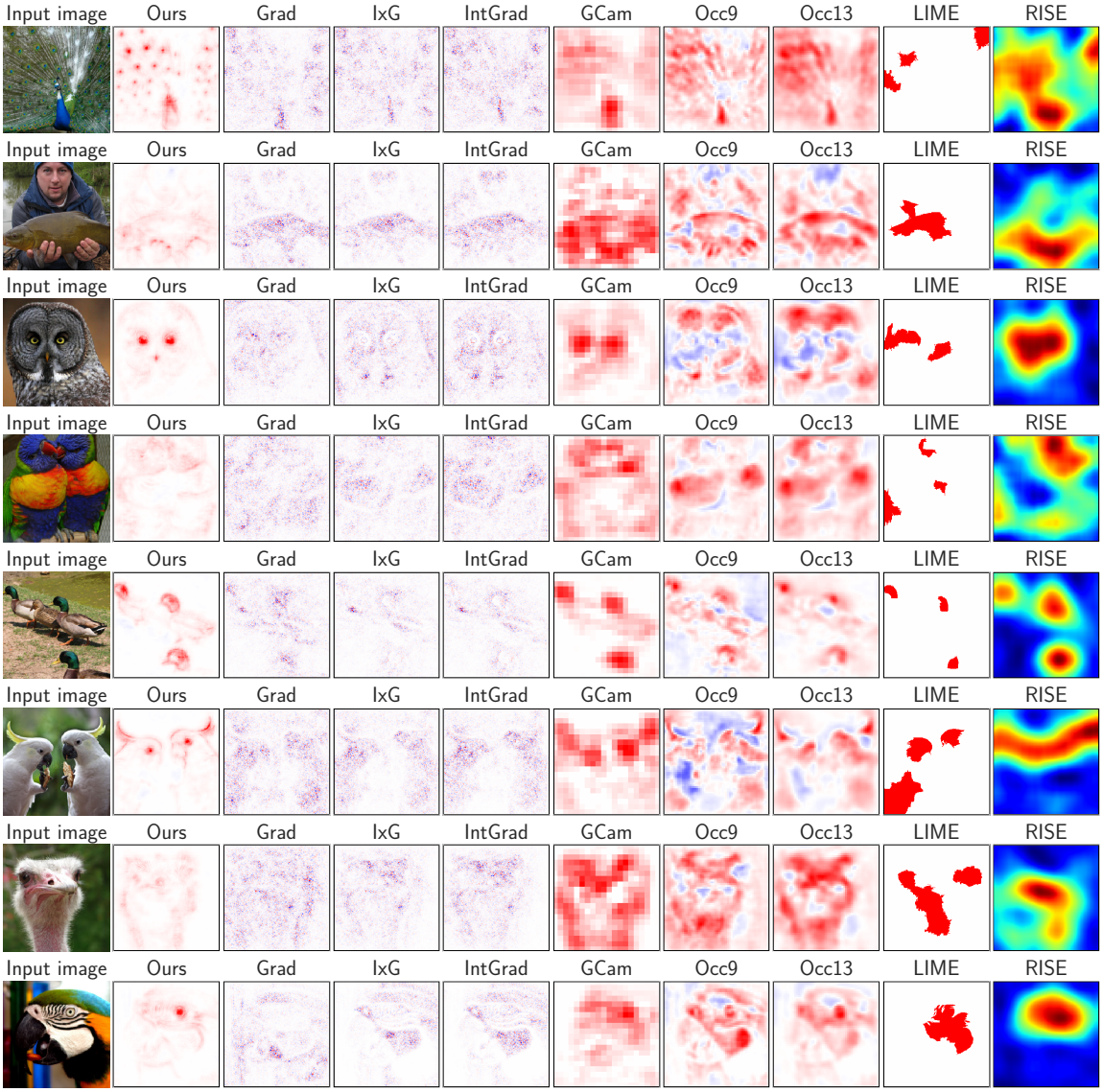


Figure B.3: Comparison between attribution methods for the most confident prediction of the first 8 classes in Figure B.1. We show positive importance attributions in red, negative attributions in blue; for RISE we use its default visualisation. Note that the model seems to align the weights well with the ornamental eyespots of the peacocks (first row, see also Figure B.1) or the heads of the ducks (row 5). While the latter are also highlighted by GCam, the former constitute a structure that is too fine-grained for GCam to resolve properly.

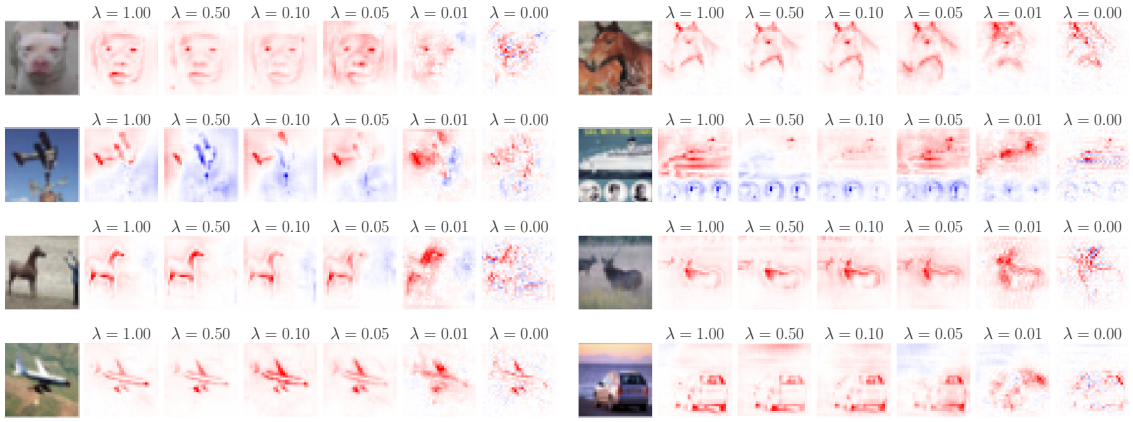


Figure B.4: Visualising the qualitative effect of explicitly regularising the linear mapping $\mathbf{W}_{0\to L}$ on CIFAR10. While the contribution maps without regularisation are noisy, they become sharper with increasing regularisation.

Regularising the Linear Mapping on CIFAR10. As mentioned in Chapter 5, the explicit representation of the model computations as a linear mapping, i.e.,

$$\hat{\mathbf{y}}(\mathbf{x}) = \mathbf{W}_{0\to L}(\mathbf{x})\mathbf{x} \quad ,$$

allows to directly regularise the linear mappings and thereby the model-inherent contribution maps. In Figure B.4 we qualitatively show how the regularisation impacts the contribution maps; for these experiments, we added a regularisation term to the loss function and optimised

$$\mathcal{L}(\mathbf{x}_i, \mathbf{y}_i) = \text{BCE}\left(\sigma(T^{-1}\mathbf{W}_{0\to L}(\mathbf{x}_i)\mathbf{x}_i + \mathbf{b}_0), \mathbf{y}_i\right) + \lambda\langle|\mathbf{W}_{0\to L}(\mathbf{x}_i)|\rangle \tag{B.1}$$

with $\langle|\mathbf{U}|\rangle$ denoting the average absolute value of a matrix $\mathbf{U}$. We see that without any regularisation the contribution maps are difficult to interpret. As soon as even a small regularisation is applied, however, the maps align well with the discriminative parts of the input image.

Sanity Check. In [AGM+18], the authors found that many commonly used methods for importance attribution are not *model-faithful*, i.e., they do not reflect the learnt parameters of the model. In order to test this, they proposed to examine how the attributions change if the model parameters are randomised step by step. If the attributions remain stable under model randomisation, they cannot be assumed to explain a specific model, but rather reflect properties of the general architecture and the input data. In Figure B.5, we show how the model-inherent contribution maps behave when re-initialising the CoDA Net layers one at a time to a random parameter setting, starting from the deepest layer. As can be seen, the contribution maps get significantly perturbed with every layer that is reset to random parameters; thus, the contribution maps pass this sanity check for attribution methods.

Contribution Maps of a Piece-wise Linear Model. In Figure B.6 we show contribution maps obtained from different pre-trained ResNet architectures obtained from https://github.com/akamaster/pytorch_resnet_cifar10. In particular, we visualise the 'Input×Gradient' method. Note that this yields contribution maps, since piece-wise linear models, such as the ResNets, produce input-dependent linear mappings, similar to the CoDA Nets. These contribution maps, however, are rather noisy and do not reveal particularly relevant features.

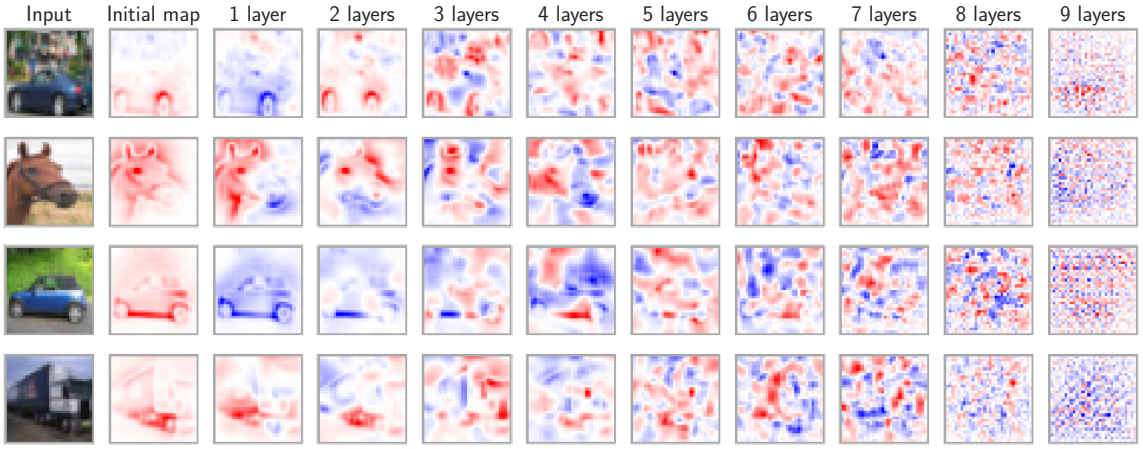


Figure B.5: Sanity check experiment as in [AGM+18]. If the importance attributions remain stable under parameter randomisation, they cannot be assumed to faithfully reflect the learnt parameters of the model. Since the contribution maps get significantly perturbed when re-initialising layers from network output (left) to network input (right), the model-inherent contribution maps thus pass this sanity check. Positive (negative) contributions shown in red (blue).

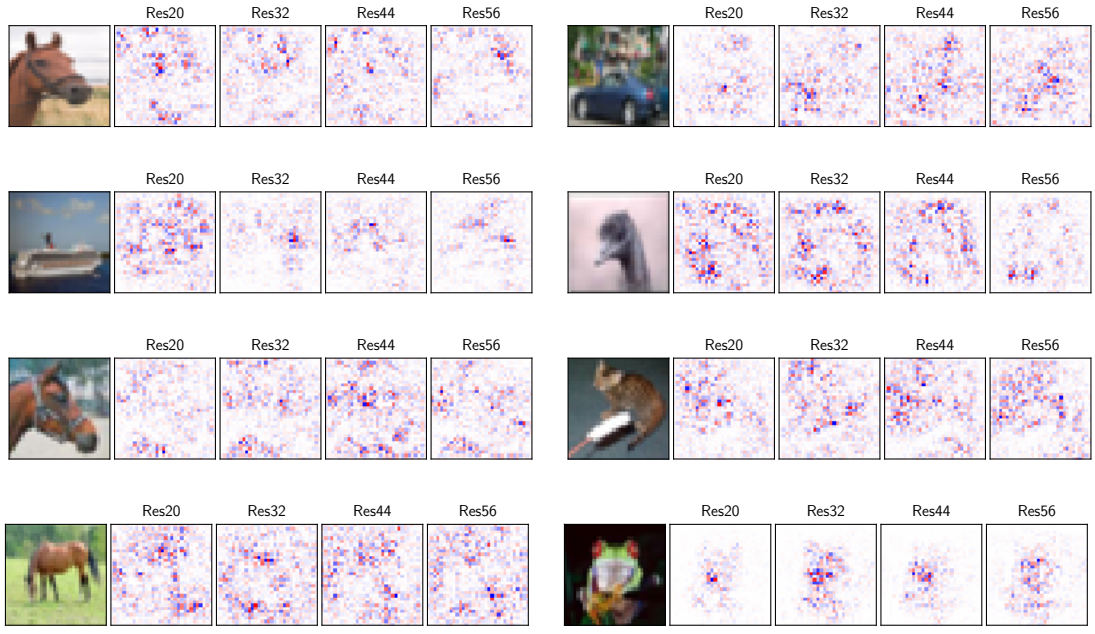


Figure B.6: 'Input×Gradient' evaluated on different ResNets. Since ResNets are piece-wise linear, and compute their output as $\mathbf{y}(\mathbf{x}) = \mathbf{M}(\mathbf{x})\mathbf{x} + \mathbf{b}(\mathbf{x})$, this is the ResNet-based equivalent to the CoDA Net contribution maps.

B.2 Additional Quantitative Results

Performance / Interpretability Trade-off. While the CoDA Nets were observed to train and perform well over a wide range of choices for the logit temperature T, there seems to be a trade-off between the accuracies of the network and their interpretability—the implicit alignment regularisation comes at a cost. For example, in Figure B.7, we contrast the gain in interpretability (left, same figure as in main paper) with the corresponding accuracies (right).

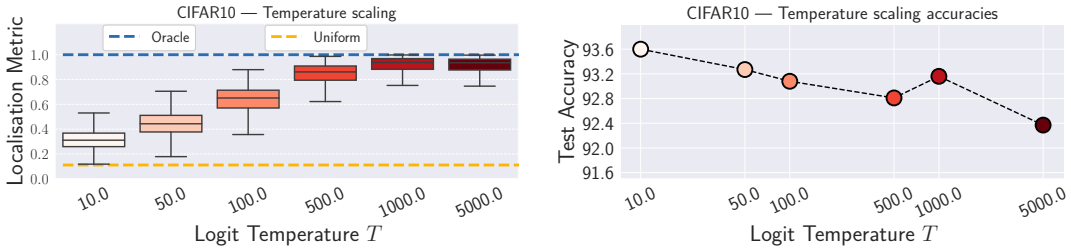


Figure B.7: **Left:** Localisation metric results for models trained with different temperatures T, same as in Chapter 5 (Figure 5.6, top right). **Right:** Corresponding accuracies of the models on the CIFAR10 test set. There seems to be a trade-off between the interpretability and the accuracy of the models due to the regularising effect of T.

Model Size vs. Accuracy. Given the quadratic form in the DAUs (cf. Equation (5.1), $\approx \mathbf{x}^T\mathbf{M}\mathbf{x}$), the number of parameters per DAU scales quadratically with the input dimensions. To limit the model size, we decided to explicitly limit the rank of the DAUs by factorising the matrix $\mathbf{M}$ into $\mathbf{AB}$ with $\mathbf{A} \in \mathbb{R}^{d\times r}$ and $\mathbf{B} \in \mathbb{R}^{r\times d}$. While this allows to be more parameter efficient, it, of course, affects the modelling capacity of the DAUs. In Figure B.8, we present how the accuracy changes with the model size; for this, we scaled the ranks of all DAUs per layer with factors of 1/8, 1/4, 1/2, and 4.5 compared to the S-CoDA-SQ model presented in Chapter 5. This results in models with 1.1M, 2.0M, 4.0M, and 34.6M parameters respectively. For comparison, the original model is also included in Figure B.8 (8M parameters).

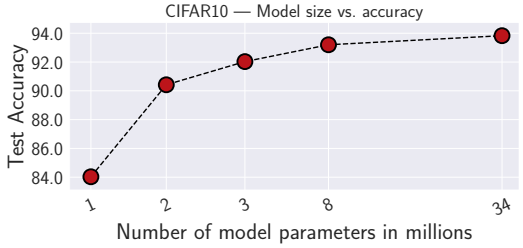
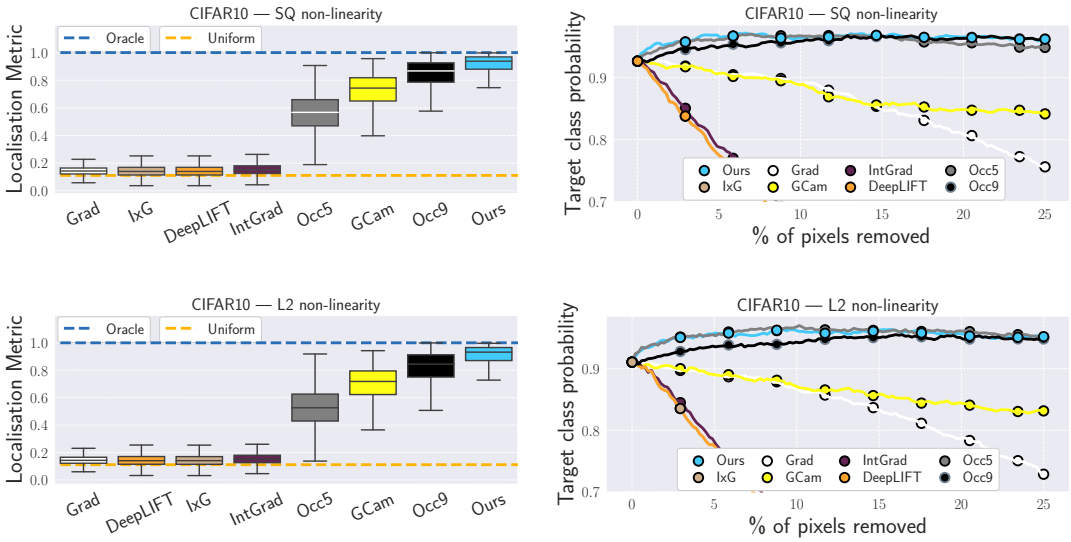


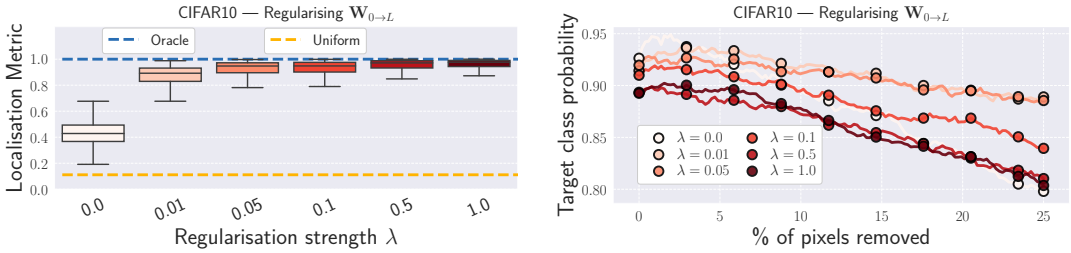
Figure B.8: Effect of scaling the rank of the DAUs in the CoDA Nets on accuracy. Specifically, all ranks in the S-CoDA-SQ model presented in Chapter 5 were scaled with the same factor, thereby changing the model size.

Interpretability Results for L2 and SQ Non-linearity. In Figure B.9a we show the results of evaluating the different methods for importance attribution on a model with the L2 and SQ non-linearities, see Equation (5.2) in Chapter 5. As can be seen, the results are very similar to those presented in Chapter 5 (Figure 5.6, centre column) for a model trained with WB rescaling; in particular, the model-inherent contribution maps outperform the other methods under the localisation metric and are on par with the

occlusion methods under the pixel removal metric; note, however, that the occlusion methods are a direct estimate of the behaviour under pixel removal and therefore expected to perform well under this metric.



(a) The quantitative results for CoDA Nets trained with the L2 and SQ non-linearities are very similar to those of a model trained with WB as shown in Chapter 5 (Figure 5.6, centre column). In particular, we observe that the CoDA Net outperforms the other methods under the localisation metric and achieves similar performance to the occlusion attribution method, which directly estimates the change in model prediction when removing a pixel.



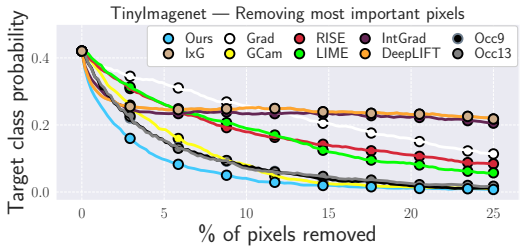
(b) Similar to the effect of temperature scaling, cf. (Figure 5.6, right column) in Chapter 5, here we show quantitative results for different regularisation strengths λ, see Equation (B.1). Similar to increasing the temperature, stronger regularisation improves localisation (left). While the pixel perturbation metric (right) also improves at first ($\lambda{=}0.01$ and $\lambda{=}0.05$), the models become less stable with stronger regularisation. In comparison, we found that increasing the temperature also improves the performance of the models under this metric (cf. Figure 5.6, right column).

Figure B.9: Quantitative results for two ablations: (a) using the L2 or SQ non-linearities and (b) increasing the regularisation of the linear mapping $\mathbf{W}_{0\to L}$, see Equation (B.1). Compare with Figure 5.6 in Chapter 5.

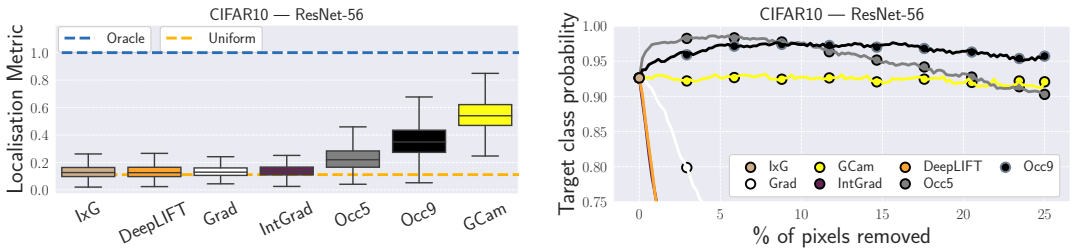
Regularisation of $\mathbf{W}_{0\to L}$. In Figure B.9b, we show the results of the localisation and the pixel removal metric for CIFAR10 models with different regularisation λ, see Equation (B.1). The localisation benefits from an increase in the regularisation strength λ. For the pixel removal metric, we observe that while the models improve at first under this metric, a strong regularisation makes the predictions more brittle.

Removing the Most Important Pixels First. In Figure B.10a, we show the results of the pixel removal metric when removing the most important pixels first. The model-inherent contributions better predict the importance of pixels and outperform the other attribution methods: the confidence drops most rapidly when removing pixels according to the ranking given by the model-inherent contribution maps.

Evaluating a Pre-trained ResNet. To establish a baseline for the performance of the different attribution methods on a classical DNN, we additionally evaluated the attribution methods on a pre-trained ResNet, see Figure B.10b. Specifically, we rely on a publicly available pre-trained ResNet-56 obtained from https://github.com/akamaster/pytorch_resnet_cifar10, which achieves a classification accuracy of 93.39%. The results show that the performance of the CoDA Net-derived contribution maps is not only strong when comparing them to attribution methods evaluated *on the same model*. Instead, they also perform well in comparison to those methods evaluated *on a different model*. In particular, the model-inherent contribution maps of the CoDA Net outperform attribution methods evaluated on the pre-trained ResNet-56 under the localisation metric. Further, only the occlusion attributions produce similarly strong pixel importance rankings for the ResNet; note, however, that the occlusion methods are a direct estimate of the behaviour under pixel removal and therefore expected to perform well under this metric.



(a) Results for the pixel removal metric when first removing the most important pixels, evaluated on the S-CoDA-SQ model. As can be seen, the ranking given by the model-inherent contribution maps seems to best reflect the pixel importance, since the confidence most rapidly drops when removing pixels according to this ranking.



(b) Quantitative results for attribution methods evaluated on a pre-trained ResNet-56 on CIFAR10. Comparing these results to Figure 5.6 (centre column) in Chapter 5 or Figure B.9a, it can be seen that the model-inherent contribution maps of the CoDA Net are also strong when compared to importance attributions evaluated on a different model.

Figure B.10: In (a) we show the results of the pixel-removal metric when removing those pixels first that are considered the most important ones according to the importance attribution method. Moreover, in (b) we plot the quantitative results for the evaluation metrics of the importance attributions for a pre-trained ResNet-56.

B.3 IMPLEMENTATION DETAILS

B.3.1 Training and Architecture Details

B.3.1.1 *Pure CoDA Networks*

Architectures. The architectures used for the results in Table 1 in Chapter 5 are given in Table B.1. All activation maps are padded with $(k-1)/2$ zeros on each side, such that the spatial dimensions are only reduced by the strides; here, k refers to the kernel size. As can be seen, the activation maps thus still have a spatial resolution after the last layer, which we further reduce with a global sum-pooling layer. Note that global sum-pooling is just a linear layer with no trainable parameters and therefore still allows for linear decomposition. For the models which use a static embedding (the image and its negative, see Section 5.2.3), the input itself consists of 6 channels; hence, the first layer takes an input with 6 channels per pixel. For the models with a learnt embedding function, the input has 32 channels. Note that for the eCoDA model the matrices $\mathbf{B}$ are not shared and each DAU has its own matrix $\mathbf{B}$.

Training Details. We use the pytorch library [PGM+19] and optimise all networks with the Adam optimiser [KB15] with default values. As for the loss function, we use the binary cross entropy loss to optimise class probabilities individually (as 'one-vs-all' classifiers). For all networks, we used a base learning rate of 2.5×10^{-4}; for the ImageNet experiment, we employed learning rate warm-up and linearly increased the learning rate from 2.5×10^{-4} to 1×10^{-3} over the first 15 (10) epochs. Further, we trained for 200 epochs on CIFAR10, for 100 epochs on TinyImageNet, and for 60 epochs on the ImageNet subset; we decreased the learning rate by a factor of 2 after every 60/30/20 epochs on CIFAR10/TinyImagenet/Imagenet; for the eCoDA experiments on CIFAR10 the learning rate was decayed according to a cosine schedule to a final learning rate of 10^{-5}. We used a batch size of 16, 128, and 64 for CIFAR10, TinyImageNet, and ImageNet respectively (and 64 for e-CoDA experiments on CIFAR10); for convergence curves of the CIFAR10 experiments, see Figure B.11. For the ImageNet subset, we additionally used RandAugment [CZSL20a] with parameters $n = 2$ and $m = 9$; for this, we relied on the publicly available implementation at https://github.com/ildoonet/pytorch-randaugment and followed their augmentation scheme. The qualitatively evaluated model (see Figures 5.7 to 5.8, 5.10 and B.1 to B.3) for the ImageNet subset was trained with $T = 1e5$ and achieved a top-1 accuracy of 76.5%. For comparison, we trained several ResNet-50 models (taken from the pytorch library [PGM+19]) with the exact same training procedure, i.e., batch size, learning rate, optimiser, augmentation, etc.). The best ResNet-50 out of 4 runs achieved 79.16% top-1 accuracy[1], which outperforms the CoDA Net but is nevertheless comparable. While it is surely possible to achieve better accuracies for both models, long training times for the CoDA Nets have thus far prevented us from properly optimising the architectures both on the 100 classes subset, as well as on the full ImageNet dataset. In order to scale the CoDA Net models to larger datasets, we believe it is important to first improve the model efficiency in future work. Lastly, when regularising the matrix entries of $\mathbf{W}_{0\to L}$, see Equation (B.1), we regularised the absolute values for the true class c, $[\mathbf{M}_{0\to L}]_c$, and a randomly sampled incorrect class per image.

B.3.1.2 *Hybrid CoDA Nets*

CIFAR10. As described in Chapter 5, the interpolation experiments for CIFAR10 are based on a ResNet-56 obtained from [Ide18]. This model consists of a convolutional layer + batch normalisation [IS15] (C+B), followed by three times nine *residual blocks* (RBs) as well as a fully connected and a pooling (FC+P) layer;

[1] The best test accuracies per run are given by 79.16%, 79.04%, 78.86%, and 78.7% respectively.

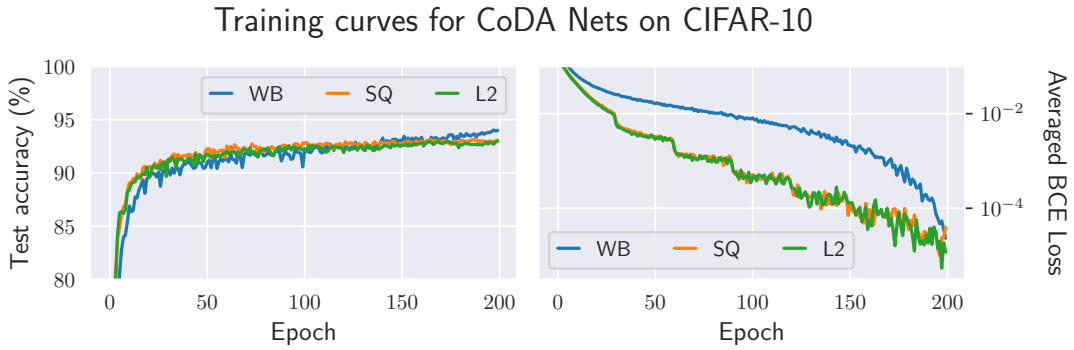


Figure B.11: Evolution of test accuracy (**left**) and the average BCE loss on the training data per epoch (**right**) as training progresses. We observe that all variants obtain competitive accuracy (>90%) already in the early stages of training, and keep improving as training continues. Note that the qualitative differences between the WB and the SQ/L2 curves reflect differences in the learning rate schedule.

for more details we kindly refer the reader to the original work [HZRS16] and the implementation [Ide18] on which we base these experiments. We can summarise this model by [C+B, 9RB, 9RB, 9RB, FC+P] and denote individual *segments* $\mathcal{S}_i$ of the model by their index in this summary counting from the back, e.g., $\mathcal{S}_5$=[C+B] and $\mathcal{S}_1$=[FC+P]. Similarly, we divide the S-eCoDA (see Table B.1) into segments, which we define as [C+B, 3eCoDA, 3eCoDA, 2eCoDA, eCoDA+P]. We 'interpolate' between these two networks by successively replacing segments from the ResNet model by its corresponding segment in the CoDA Net—each replacement yields one hybrid model, which we train on the fixed representations[2] obtained from the ResNet-based stem. We use the same training scheme as in the S-eCoDA-based models on CIFAR10 without learning rate warm-up. Further, the models with $[1, 2, 3, 4]$ replaced segments are trained with temperatures of $10^{[1,3,5,5]}$.

Imagenet. The interpolation experiments on the ImageNet dataset are based on a pre-trained ResNet-50 obtained from the pytorch [PGM+19] library. Here, we define individual segments by the final classification layer + pooling and individual ResNet bottlneck blocks. For the corresponding eCoDA Net, the segments contain exactly one eCoDA layer. We train with a batch size of 128, DAU ranks of 8, 100 epochs, a cosine learning rate decay from 10^{-3} to 10^{-5}, a temperature $T = 100$ and apply RandAugment [CZSL20a] with default parameters on random crops of size 192×192.

B.3.2 Convolutional Dynamic Alignment Units

In Algorithm B.1, we present the implementation of the convolutional DAUs (CoDAUs). As can be seen, a Convolutional Dynamic Alignment Layer applies dynamic (input-dependent) filters to each of the patches extracted at different spatial locations. In detail, the Dynamic Alignment Units are implemented as two consecutive convolutions (lines 11 and 15), which are equivalent to first applying matrix $\mathbf{B}$ (line 24) and then $\mathbf{A}$ to each patch and adding a bias term $\mathbf{b}$ (line 29). After applying the non-linearity (line 33), we obtain the dynamic weight vectors for CoDAUs as described in Equation (5.1) in Chapter 5. In particular, for every patch $\mathbf{p}_{hw}$ extracted at the spatial positions hw in the input, we obtain the dynamic weight $\mathbf{w}_j(\mathbf{p}_{hw})$ for the j-th DAU as

$$\mathbf{w}_j(\mathbf{p}_{hw}) = g(\mathbf{A}_j\mathbf{B}\mathbf{p}_{hw} + \mathbf{b}_j) \; ; \tag{B.2}$$

[2] I.e., the ResNet weights are fixed in these experiments.

note that the projection matrices $\mathbf{B}$ are thus shared between the DAUs. These weights are then applied to the respective locations (line 41) to yield the outputs of the DAUs per spatial location. As becomes apparent in line 41, the outputs are linear transformations (weighted sums) of the input and can be written as

$$\mathbf{a}_{l+1}(\mathbf{a}_l) = \mathbf{W}(\mathbf{a}_l)\mathbf{a}_l \quad , \tag{B.3}$$

with $\mathbf{a}_l \in \mathbb{R}^d$ the vectorised input to layer l and $\mathbf{W} \in \mathbb{R}^{f \times d}$ and f the number of filters (DAUs). The rows in matrix $\mathbf{W}$ correspond to exactly one filter (DAU) applied to exactly one patch $\mathbf{p}_{hw}$ and are non-zero only at those positions that correspond to this specific patch in the input.

B.3.3 Attribution Methods

In Section 5.3.2, we compare the model-inherent contribution maps of the CoDA Nets to those of the following methods for importance attribution: the gradient of the class logits with respect to the input image [BSH+10] (Grad), 'Input×Gradient' (IxG, cf. [AGM+18]), GradCAM [SCD+17] (GCam), Integrated Gradients [STY17] (IntG), DeepLIFT [SGK17], several occlusion sensitivities (Occ-K, with K the size of the occlusion patch) [ZF14], RISE [PDS18], and LIME [RSG16].

For RISE and LIME, we relied on the official implementations available at https://github.com/eclique/RISE and https://github.com/marcotcr/lime respectively. For RISE, we generated 6000 masks with parameters $s = 6$ and $p_1 = 0.1$. For LIME, we evaluated on 256 samples per image and used the top 3 features for the attribution; for the segmentation, we also used the default parameters, namely 'quickshift' with $max\_dist = 200$, $ratio = 0.2$, and a kernel size of 4.

For Grad, GCam, IxG, IntG, DeepLIFT, and the occlusion sensitivities, we relied on the publicly available pytorch library 'captum' (https://github.com/pytorch/captum). GCam was used on the last activation map before global sum-pooling. The occlusion sensitivities were used with strides of 2 on CIFAR10 and strides of 4 for TinyImageNet. Finally, for IntG we used 20 steps for the integral approximation.

B.3.4 Evaluation Metrics

In Section 5.3.2, we evaluated the attribution methods against 2 quantitative metrics: **(1)** the adapted *pointing game* [ZBL+18] and **(2)** the prediction stability under removing the *least important pixels* as in [SF19]. In section B.2, we further show results for removing the *most important pixels* first.

For **(1)**, we constructed 500 (250) 3×3 multi-images for CIFAR10 (TinyImageNet); for an example with 2×2, see Fig. 7 in Chapter 5. In each of these multi-images, every class occurred at most once. As stated in Section 5.3.2, we measured the fraction of positive contributions falling inside the correct mini-image. Further, the images were sorted according to their confidence for each of the classes. For every multi-image, a random set of classes was sampled. For each of the classes, we included the most confidently classified class image in the multi-image that had not been used yet in previous multi-images.

For **(2)**, we followed [SF19] and successively replaced one (embedded) pixel at a time, replacing its embedding by a corresponding zero vector, until up to 25% of the image were removed. The pixels were removed in order, sorted by their assigned importance.

Network	Input dimensions	Layer	Number of DAUs	Rank of AB	Kernel size	Stride
S-CoDA	$6 \times 32 \times 32$	1	16	32	3	1
		2	16	32	3	1
		3	32	64	3	2
		4	32	64	3	1
		5	32	64	3	1
		6	64	64	3	2
		7	64	64	3	1
		8	64	64	3	1
		9	10	64	1	1
S-eCoDA	$6 \times 32 \times 32$	1	16	16	3	1
		2	16	16	3	1
		3	32	16	3	2
		4	32	16	3	1
		5	32	32	3	1
		6	64	32	3	2
		7	64	32	3	1
		8	64	32	3	1
		9	10	32	1	1
L-CoDA	$6 \times 240 \times 240$	1	16	64	7	3
		2	32	64	3	1
		3	32	64	3	1
		4	64	128	3	2
		5	64	128	3	1
		6	64	128	3	1
		7	64	256	3	2
		8	64	256	3	1
		9	100	256	3	1
XL-CoDA	$6 \times 64 \times 64$	1	16	64	5	1
		2	32	64	3	1
		3	32	128	3	2
		4	64	128	3	1
		5	64	128	3	1
		6	64	256	3	2
		7	64	256	3	1
		8	64	256	3	1
		9	200	256	3	2

Table B.1: Architecture details for the results presented in Table 5.1 in Chapter 5.

Algorithm B.1: Implementation of a Convolutional Dynamic Alignment Layer

```python
from torch import nn
import torch.nn.functional as F


class DAUConv2d(nn.Module):

    def __init__(self, in_channels, out_channels, rank, kernel_size, stride, padding, act_func):
        # act_func: non-linearity for scaling the weights. E.g., L2 or SQ.
        # out_channels: Number of convolutional DAUs for this layer.
        # rank: Rank of the matrix AB.
        # 'dim_reduction' applies matrix B.
        self.dim_reduction = nn.Conv2d(in_channels, rank, kernel_size, stride, padding, bias=False)
        # Total dimensionality of a single patch
        self.patch_dim = in_channels * kernel_size * kernel_size
        # 'weightings' applies matrix A and adds bias b.
        self.weightings = nn.Conv2d(rank, out_channels * self.patch_dim , kernel_size=1, bias=True)
        self.act_func = act_func
        self.out_channels = out_channels
        self.kernel_size = kernel_size
        self.stride = stride
        self.padding = padding

    def forward(self, in_tensor):
        # Project to lower dimensional representation, i.e., apply matrix B. This yields Bp for every patch p.
        reduced = self.dim_reduction(in_tensor)
        # Get new spatial size height h and width w
        h, w = reduced.shape[-2:]
        batch_size = in_tensor.shape[0]
        # Apply matrix A and add bias b, yielding ABp + b for every patch p.
        weights = self.weightings(reduced)
        # Reshape for every location to size patch_dim×out_channels
        weights = weights.view(batch_size, self.patch_dim, out_channels, h, w)
        # Apply non-linearity to the weights, yielding w(p) = g(ABp + b) as in Equation (5.1) for every patch p.
        weights = self.act_func(weights, dim=1)
        # Extract patches from the input to apply dynamic weights to patches.
        patches = F.unfold(in_tensor, self.kernel_size, padding=self.padding, stride=self.stride)
        # Reshape for applying weights.
        patches = patches.view(batch_size, self.patch_dim, 1, h, w)
        # Apply the weights to the patches.
        # As can be seen, the output is just a weighted combination of the input, i.e., a linear transformation.
        # The output can thus be written as o = W(x)x.
        return (patches * weights).sum(1)
```

B.4 Comparison to Capsule Networks

To discuss the relationship to capsule networks, in section B.4.1 we will first rewrite the classical capsule formulation in 'Dynamic Routing Between Capsules' by Sabour et al. [SFH17] to mitigate the notational differences. In section B.4.2 we then show that while capsules and Dynamic Alignment Units share some computations, there are several important differences that we summarise in Table B.2.

	Classical capsules	**Dynamic Alignment Units**
Non-Lin. g	SQ	SQ, L2, ...
Activations	$g\left(\mathbf{V}\mathbf{x}\right)$	$g\left(\mathbf{A}\mathbf{B}\mathbf{x}+\mathbf{b}\right)$
Routing	**yes**	**no**
Low-rank	**no**	**yes**
Output	$\mathrm{CAP}(\mathbf{x})$	$\mathrm{CAP}(\mathbf{x})^T\mathbf{x}$

Table B.2: Comparison between capsules and Dynamic Alignment Units (DAUs). Importantly, DAUs produce a linear transformation of the input by multiplying the 'capsule activations' with the input (see 'Output') and allow for constraining the rank of the transformation. The dynamic weights in the DAUs can be seen as the activations of a capsule, such that $\mathbf{w}(\mathbf{x}) = \mathrm{CAP}(\mathbf{x})$, see 'Output' in the table.

B.4.1 Reformulating Capsules

In this subsection, we will show that the classical capsule formulation (Equation (B.4)), in which input capsules 'vote' for the activations of an output capsule, can be written as a simple linear transformation $\mathbf{s} = \mathbf{V}\mathbf{x}$ if just one iteration of the dynamic routing algorithm is applied; here, $\mathbf{s}$ is a vector containing the activations of the output capsule, $\mathbf{V}$ stores the 'votes' of the input capsules to an output capsule, and $\mathbf{x}$ is a vector containing the activations of all input capsules. In the following, we will start from how capsules are formulated in [SFH17] and rewrite this formulation step by step.

In [SFH17] Equation (2), the authors calculate the activations $\mathbf{s}$ of a capsule[3] *before* any routing as

$$\mathbf{s} = \sum_i c_i \hat{\mathbf{u}}_i \ , \quad \hat{\mathbf{u}}_i = \mathbf{W}_i \mathbf{u}_i \quad . \tag{B.4}$$

Here, $\mathbf{u}_i$ is the vector of capsule i from the incoming layer and $\mathbf{W}_i$ a matrix which transforms the capsule activations to generate the votes $\hat{\mathbf{u}}_i$, i.e., the vote of the i-th incoming capsule to the output capsule in the current layer. Note that c_i are the coefficients for the dynamic routing algorithm. If no routing is applied, they can be combined with $\mathbf{W}_i$ to yield $\widetilde{\mathbf{W}}_i = c_i \mathbf{W}_i$, which simplifies Equation (B.4) to

$$\mathbf{s} = \sum_i \hat{\mathbf{u}}_i \ , \quad \hat{\mathbf{u}}_i = \widetilde{\mathbf{W}}_i \mathbf{u}_i \quad . \tag{B.5}$$

We note that the linear transformation of $\mathbf{u}_i$ can be represented as votes of the individual entries t of $\mathbf{u}_i$:

$$\hat{\mathbf{u}}_i = \widetilde{\mathbf{W}}_i \mathbf{u}_i = \sum_t \left[\mathbf{u}_i\right]_t \left[\widetilde{\mathbf{W}}_i^T\right]_t \quad . \tag{B.6}$$

[3] As we only discuss a single output capsule, we omitted the subscript j for the j-th output capsule for simplicity.

Here, $[\mathbf{W}]_t$ denotes the t-th row in a matrix $\mathbf{W}$ (equivalently for a vector). Inserting this formulation of $\hat{\mathbf{u}}_i$ in Equation (B.5) yields

$$\mathbf{s} = \sum_i \sum_t [\mathbf{u}_i]_t \left[\widetilde{\mathbf{W}}_i^T\right]_t \quad . \tag{B.7}$$

Hence, $\mathbf{s}$ can be represented as the result of votes from all neurons u contained in any of the incoming capsules (note that we sum over all entries in all capsules). If we represent the activations of these neurons in a single vector $\mathbf{x}$, denote their activations by x_u, and their respective votes for the output capsule by $\mathbf{v}_u$ we can write Equation (B.7) as:

$$\mathbf{s} = \sum_i \sum_t \overbrace{[\mathbf{u}_i]_t}^{\widehat{=}x_u} \overbrace{\left[\widetilde{\mathbf{W}}_i^T\right]_t}^{\widehat{=}\mathbf{v}_u} = \sum_u x_u \mathbf{v}_u \quad . \tag{B.8}$$

The sum on the right hand side in Equation (B.8), in turn, can be expressed as a simple matrix-vector multiplication, such that

$$\mathbf{s} = \sum_u x_u \mathbf{v}_u = \mathbf{V}\mathbf{x} \tag{B.9}$$

with $\mathbf{v}_u$ as the columns of $\mathbf{V}$. The result is, of course, trivial and only shows that the capsule activations (a weighted sum of the inputs, Equation (B.4)) are obtained as a linear transformation of the input if no dynamic routing is applied.

Finally, we note that subsequent to this linear transformation, [SFH17] apply the squashing function SQ (see Equation (5.2) in Chapter 5) to the output vector $\mathbf{s}$, which yields the final capsule output $\text{CAP}(\mathbf{x})$:

$$\text{CAP}(\mathbf{x}) = \text{SQ}(\mathbf{s}(\mathbf{x})) = \text{SQ}(\mathbf{V}\mathbf{x}) \quad . \tag{B.10}$$

B.4.2 Differences to Capsules

In the previous section we showed that the activation of a single capsule is computed as a linear transformation of all input activations, which is subsequently squashed (see Equation (B.10)). Comparing this with the computation of the DAU outputs (Equation (5.1)), we see that this is equivalent to the dynamic weight computation if $\mathbf{V}=\mathbf{A}\mathbf{B}$ and $\mathbf{b}=\mathbf{0}$. As such, Dynamic Alignment Units and capsules are related. However, there are important differences, which we discuss in the following and summarise in Table B.2.

First, in [SFH17], the squashed capsule activations $\text{CAP}(\mathbf{x})$ are used as input to the next layer (after potentially applying the dynamic routing algorithm first). Instead of forwarding the squashed activations directly, in our DAUs they are used to linearly transform the input. Second, by factorising the matrix $\mathbf{V}$ into two matrices $\mathbf{A}\mathbf{B}$, we are able to control the rank of the linear transformation. Third, when computing the weights in the DAUs we allow for an additional bias term, which in the context of capsules can be considered a 'default vote'. Fourth, we generalise the non-linearity in the DAUs to any non-linearity that only changes the norm. Lastly, we do not apply dynamic routing in the DAUs.

C

APPENDIX — B-COS NETWORKS

In this supplement to our work on B-cos DNNs, we provide:

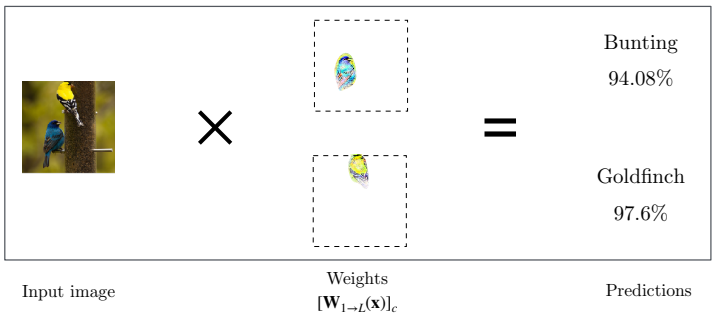


Figure C.1: **Illustration of the computations of a B-cos Network**. For a given input image (*left*), the model computes an *input-dependent* linear transform $\mathbf{W}_{1\to L}(\mathbf{x})$ (*centre*). The scalar product between the input and the weights $[\mathbf{W}_{1\to L}(\mathbf{x})]_c$ for class c (row c of $\mathbf{W}_{1\to L}(\mathbf{x})$), yields the class logits for the respective class. To obtain class probabilities (*right*), we apply the sigmoid function. Since the B-cos Networks are trained with the BCE loss, they produce probabilities *per class* and *not a probability distribution over classes*. Thus, the probabilities do not sum to 1. For illustration purposes, we only visualise the positive contributions according to $\mathbf{W}_{1\to L}(\mathbf{x})$.

C.1 ADDITIONAL QUALITATIVE EXAMPLES

In Figure C.1, we illustrate how the linear mappings $\mathbf{W}_{1\to L}(\mathbf{x})$ are used to compute the outputs of B-cos Networks. In particular, with this we would like to highlight that these linear mappings do not only constitute qualitatively convincing visualisations. Instead, they are in fact based on the actual linear transformation matrix that the model effectively applies to the input to compute its outputs and thus constitute an accurate summary of the model computations.

Comparison between Model Explanations. In Figures C.2 and C.3, we compare explanations for different network architectures on the same set of images. We find that all models generally seem to focus on the same regions in the image. However, we also observe architecture-specific details: e.g., we find that the vision transformer yield much sparser explanations than the convolutional models and ResNet models (ResNet-152 and ResNext-50 in the figures) exhibit grid-pattern artefacts, which we attribute to the downsampling mechanism in those models. Considering that information is not processed as locally in the ViT$_C$ models as in the convolutional models, these qualitative results provide further evidence for the model-faithfulness of the explanations based on $\mathbf{W}(\mathbf{x})$.

Figure C.2: **Additional model explanations** based on $\mathbf{W}(\mathbf{x})$ for various B-cos architectures, see also Figure C.3. While the explanations are generally consistent between models, architecture-specific details can be observed. E.g., ResNet-based architectures exhibit a grid-pattern in the explanations, and we find that the ViT models generally produce sparser explanations.

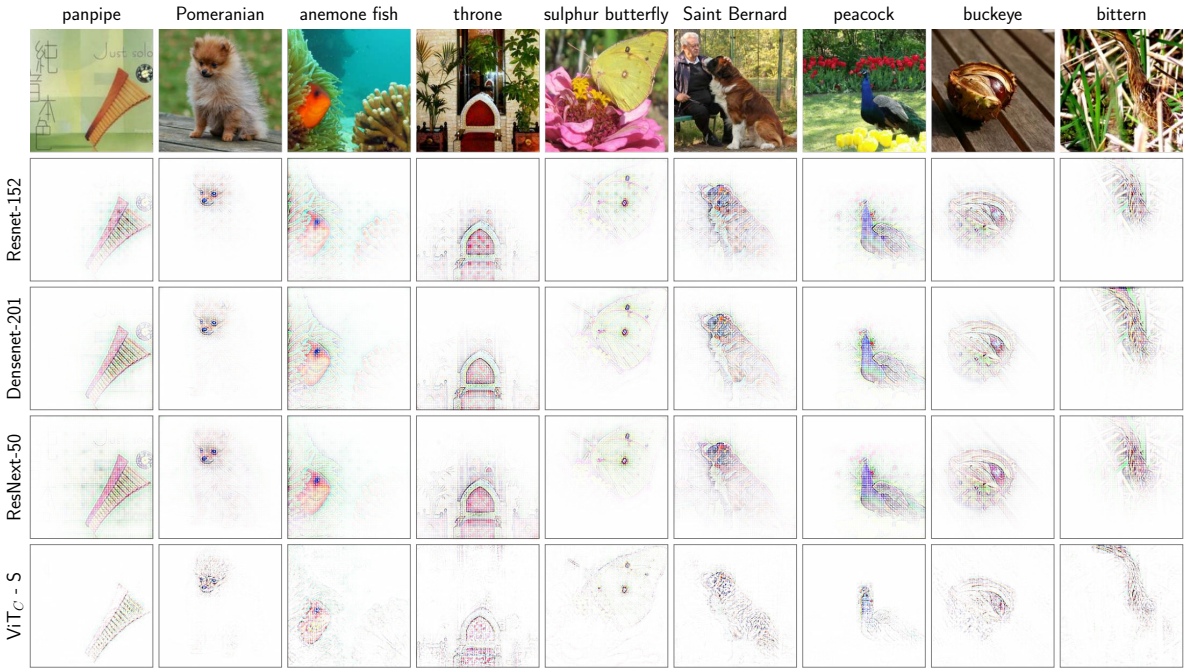


Figure C.3: **Additional model explanations** based on $\mathbf{W}(\mathbf{x})$ for various B-cos architectures, see also Figure C.2. While the explanations are generally consistent between models, architecture-specific details can be observed. E.g., ResNet-based architectures exhibit a grid-pattern in the explanations, and we find that the ViT models generally produce sparser explanations.

C.2 Additional Quantitative Evaluations

In Figure C.4, we present the localisation results of the grid pointing game [BFS21] for two additional model architectures, see also Figure 6.6 in Chapter 6.

In particular, in Figure C.4 (left), we show that of all methods, the best explanation for the B-cos models is given by the model-inherent linear transformations $\mathbf{W}_{1\to L}(\mathbf{x})$. Moreover, in Figure C.4 (right) we can see that the B-cos explanations yield a significant interpretability gain over the respective baselines: specifically, we see that no method explains the baseline models better than the model-inherent linear transformations explain the respective B-cos Network.

Finally, as discussed in Chapter 6 (Figure 6.7), the results of the B-cos explanations improve further by including only the top-n pixels (here $\approx$11.3k) in the explanations, denoted by OursQ in the figure.

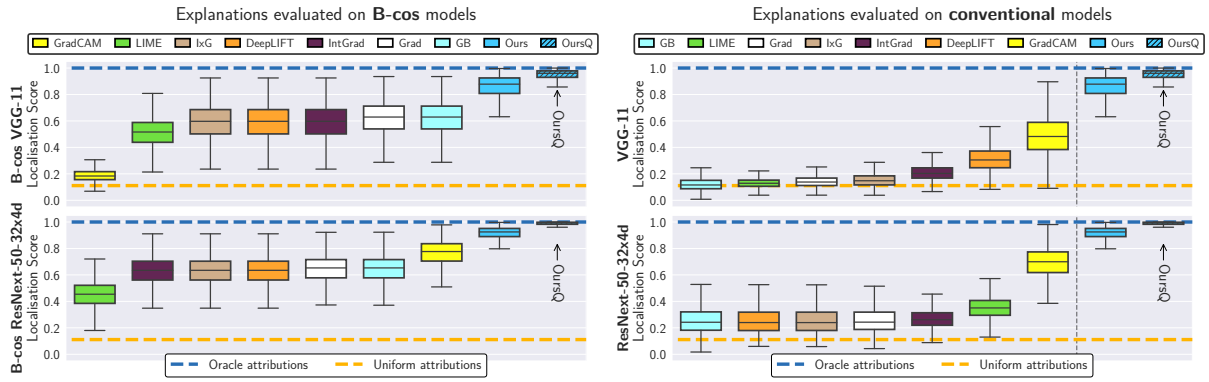


Figure C.4: **Additional localisation results.** We show localisation results of two additional B-cos models (left) and the conventional counterparts (right). These results are consistent with those shown in Figure 6.6. Specifically, we find the model-inherent explanations (Ours) of the B-cos models to yield close to optimal localisation results, especially when only including the top 11.3k pixels in the explanations (OursQ). Additionally, and in accordance with the results reported in Chapter 6 for a DenseNet and a ResNet model, we find that the B-cos models are better explained by post-hoc explanation methods than the conventional models, except for GradCAM on VGG. For VGG, the gradient computation of GradCAM seems to be significantly affected by the non-linear aspects of the B-cos transformations (note that VGG is the only model using a multi-layer perceptron for the classification head).

C.3 Implementation Details

In the following, we provide further implementation details regarding implementation of a convolutional B-cos transform (Algorithm C.1), the training and evaluation procedure (Section C.3.1), and the post-hoc attribution methods (Section C.3.2).

Algorithm C.1: Pseudocode for B-cos-Conv2d, cf. Equation (6.10) in Chapter 6.

```
# x: input, Ŵ: normed weights, k: kernel size, d_f: index of feature dimension
def bcos_conv2d(x, Ŵ, k, B):
    linear_out = conv2d(x, Ŵ) # = Ŵx
    norm = sumpool2d(x.pow(2).sum(d_f), k).sqrt()
    cos = linear_out / norm.unsqueeze(d_f)
    scaling = cos.abs().pow(B-1) # = |c(x; Ŵ)|^{B-1}
    return scaling * linear_out # = |c(x; Ŵ)|^{B-1} Ŵx
```

C.3.1 Training and Evaluation Procedure

C.3.1.1 *CIFAR10*

Architecture. For our simple B-cos models trained on CIFAR10, we used a 9-layer architecture with the following specifications: kernel size $k = [3, 3, 3, 3, 3, 3, 3, 3, 1]$, stride $s = [1, 1, 2, 1, 1, 2, 1, 1, 1]$, padding $p = [1, 1, 1, 1, 1, 1, 1, 1, 0]$, and output channels $o = [64, 64, 128, 128, 128, 256, 256, 256, 10]$ for layers $l = [1, 2, 3, 4, 5, 6, 7, 8, 9]$ respectively.

When increasing the parameter B, we observed the input signal to decay strongly over the network layers,

which resulted in zero outputs and hindered training. To overcome this, we scaled all layer outputs with a fixed scalar γ to improve signal propagation, which we set such that $\log_{10}\gamma{=}1.75 + B/10$.. To counteract the artificial upscaling of the signal at the network output, we down-scaled the network output by a fixed constant T for each B, such that $\log_{10} T = [8.9, 8.125, 7.35, 6.757, 4.8, 4.525, 4.25]$ for $\mathrm{B} = [1, 1.25, 1.5, 1.75, 2, 2.25, 2.5]$ respectively.

Furthermore, for our B-cos ResNet models for CIFAR10 we converted the ResNet-{20,32,44,56} [HZRS16] models to their corresponding B-cos version. Specifically, to evaluate the applicability of different normalisation layers, we replaced the conventionally used Batch-Norm layers by Batch-, Layer-, Instance-, Position-, and All-Norm layers (Equations (6.20), (6.21), (6.23) and (6.24)). As we report in Figure 6.13 in Chapter 6, for some of the normalisation layers (e.g. All-, Instance-, and Batch-Norm), the bias terms can play a significant role in the classification decision. Therefore, for the ImageNet experiments, we exclusively use *bias-free* normalisation layers. Furthermore, we switch the order of the the global pooling layer and last linear classifier layer, *i.e.* we first do a linear classification (using a convolution) before doing a global (average) pooling, in order to more easily evaluate those models in the localisation metric, for which the size of the input image changes. Note that for models with a linear classification head, this yields functionally equivalent models; for the B-cos models, however, the ordering matters, since the classification head is a non-linear (i.e. B-cos) layer.

Training. We trained all our CIFAR10 models for 100 epochs with the Adam optimiser [KB15], an initial learning rate of 1×10^{-3}, and a batch size of 64. Further, we used a cosine learning rate schedule and decayed the learning rate to 1×10^{-5} for our simple B-cos models and to 0 for our B-cos ResNets over 100 epochs. For augmenting the data, we applied horizontal flipping and padded random cropping. We used a bias term of $\mathbf{b}{=}\log(0.1/0.9)$, which yields a uniform probability distribution for zero inputs ($[\mathbf{f}(\mathbf{x}=\mathbf{0})]_i{=}[\sigma(\mathbf{W}_{1\to L}\,\mathbf{0}+\mathbf{b})]_i{=}0.1\;\forall\; i$).

C.3.1.2 *ImageNet*

Training. Our training recipe is based on the `torchvision` (https://github.com/pytorch/vision/) reference recipes. For our ResNets, DenseNets, and VGGs, we train for 90 epochs with the Adam optimiser using a learning rate of $1 \times 10{-}3$ and a batch size of 256 (distributed over 4 GPUs with each having batch size of 64). Following `torchvision`, we also train our ResNeXt model with the same recipe as for the ResNets but for 100 epochs instead of 90. As mentioned in Section 6.1.2.2 in Chapter 6, we change the target encoding $\mathbf{y}_i$ for non-target classes to be $1/C$. For the learning rate schedule, we first use a linear warm-up for 5 epochs, then we use a cosine annealing learning schedule, decaying the learning rate to 0. To avoid divergence issues we employ gradient clipping. However, we empirically noticed that the magnitude of the gradient for B-cos models tends to be significantly lower than that of the gradients for standard networks. To avoid manually tuning the clipping threshold, we opt for employing Adaptive Gradient Clipping (AGC) [BDSS21a]. For data augmentation, we use random cropping and then resizing with bilinear interpolation to size 224 along with random horizontal flips with a probability of 0.5.

Inspired by `torchvision`'s new longer training recipes[1], we additionally train a set of ConvNeXt-{Tiny, Base}, DenseNet-121, and ResNet {50, 152} models for 600 epochs with the Adam optimiser using a learning rate of 1×10^{-3} and a batch size of 1024 (distributed over 8 GPUs with each having a batch size of 128). We keep an exponential moving average (EMA) of the weights of the model with a decay rate of 0.99998 updated after every 32 training steps. We use the same learning rate scheduler as mentioned above and AGC for gradient clipping. For data augmentation, we use random cropping and then resizing to size 176 with bilinear interpolation, MixUp [ZCDLP18] with $\alpha = 0.2$, CutMix [YHO+19] with $\alpha = 1.0$,

[1] https://pytorch.org/blog/how-to-train-state-of-the-art-models-using-torchvision-latest-primitives/

Repeated Augmentations [BJV+19, HBH+20] with 4 repetitions, Random Erasing [ZZK+20, DT17] with a probability of 0.1, horizontal flipping with a probability of 0.5, and finally TrivialAugment [MH21] which is parameter-free.

We trained our ViTs from scratch following the 90 epoch protocol described by [BZK22]. Given that all the B-cos weights are normalised to unit norm, we employ Adam instead of AdamW, since no weight decay is necessary. Furthermore, as mentioned previously we employ AGC over standard gradient clipping. We used an effective batch size of 1024 (distributed over 8 GPUs) for all but our largest ViTs (ViT-L models and the Simple ViT-B model), for which we use a batch size of 512 due to memory constraints. We observed these larger ViT models to diverge at the beginning of training and therefore opted to increase the length of the warm-up period from the 10 000 steps described by [BZK22] to 50 000 steps.

For all further details, we kindly refer the reader to the code at https://github.com/B-cos/B-cos-v2.

Evaluation. We evaluated all networks on the full images of the ImageNet validation set, after rescaling the smaller dimension (height / width) to 256 using bilinear interpolation. For networks with EMA weights, we evaluate both the EMA weights non-EMA weights and report the best performing one.

C.3.2 Attribution Methods

We compare the model-inherent explanations, given by the linear transformation $\mathbf{W}_{1\to L}(\mathbf{x})$, against the following post-hoc attribution methods: the vanilla gradient (Grad, [BSH+10]), 'Input×Gradient' (IxG, [SGK17]), Guided Backpropagation' (GB, [SDBR15]), Integrated Gradients (IntGrad, [STY17]), DeepLIFT ([SGK17]), GradCAM ([SCD+17]), and LIME ([RSG16]).

For all methods except LIME, we rely on the captum library (github.com/pytorch/captum). For IntGrad, we set n_steps $= 50$ for integrating over the gradients. For LIME, we used the official implementation available at github.com/marotcr/lime, with 500 samples and the default values for the kernel size ($k = 4$) and evaluate the weights assigned to the individual image segments, see Figure 6.9 for a visualisation.

C.3.2.1 *Localisation Metric*

We evaluated all attribution methods on the grid pointing game [BFS21]. For this, we constructed 500 3×3 grid images. For an example of a 2×2 grid, see Figure 6.3 in Chapter 6. As was done in [BFS21], we sorted the images according to the models' classification confidence for each class and then sampled a random set of classes for each multi-image. For each of the sampled classes, we then included the most confidently classified image in the grid that had not already been used in a previous grid image.

C.4 ADDITIONAL DERIVATIONS AND DISCUSSIONS

C.4.1 On the Relation of B-cos to RBF Networks

In contrast to Deep Neural Networks (DNNs), Radial Basis Function (RBF) networks [BL88] typically consist of only a single hidden layer [HW19], in which unit i is modelled as:

$$\rho(\mathbf{x}; \mathbf{w}_i) = \exp\left[-\|\mathbf{x} - \mathbf{w}_i\|^2/\sigma\right] . \tag{C.1}$$

Here, σ denotes a scalar hyperparameter and $\mathbf{w}_i$ the learnable weight vector of the i-th unit. While DNNs have proven more successful on complex tasks, RBF Networks exhibit other desirable properties: e.g.,

they have been shown to be more robust to adversarial examples [GSS15] and to inherently provide low confidence predictions far from the training data [GSS15, HAB19].

Despite this, relatively few attempts have been made to leverage the benefits of RBF Networks within deep networks [HW19, ZHS18]. Interestingly, we find that B-cos Networks exhibit certain parallels to RBF Networks, which would be interesting to further investigate in future work.

In particular, let us consider the following function h, in which the output of a linear unit ($\mathbf{w}^T\mathbf{x}$) is 'gated' by an RBF unit with the same parameters $\mathbf{w}$:

$$h(\mathbf{x};\mathbf{w}) = \alpha \times \mathbf{w}^T\mathbf{x} \times \left(\exp\left[-\|\widehat{\mathbf{x}} - \widehat{\mathbf{w}}\|^2/2\sigma\right] - b\right) \tag{C.2}$$

Here, α is a scalar hyperparameter and a bias b is added to the exponential function for convenience (see below). Moreover, $\widehat{\mathbf{v}} = \mathbf{v}/\|\mathbf{v}\|$, i.e., the hat operator denotes that a given vector $\mathbf{v}$ is of unit norm. As such, we can rewrite the function h as follows:

$$h(\mathbf{x};\mathbf{w}) = \alpha \times \mathbf{w}^T\mathbf{x} \times \left(\exp\left[-\|\widehat{\mathbf{x}} - \widehat{\mathbf{w}}\|^2/2\sigma\right] - b\right) \tag{C.3}$$

$$\text{(compute norm)} \quad = \alpha \times \mathbf{w}^T\mathbf{x} \times \left(\exp\left[-(\underbrace{\widehat{\mathbf{x}}^T\widehat{\mathbf{x}}}_{=1} + \underbrace{\widehat{\mathbf{w}}^T\widehat{\mathbf{w}}}_{=1} - 2\widehat{\mathbf{w}}^T\widehat{\mathbf{x}})/2\sigma\right] - b\right) \tag{C.4}$$

$$\text{(pull out constants)} \quad = \underbrace{(\alpha \times \exp\left[-1/\sigma\right])}_{\alpha'} \times \mathbf{w}^T\mathbf{x} \times \left(\exp\left[\widehat{\mathbf{w}}^T\widehat{\mathbf{x}}/\sigma\right] - \underbrace{b \times \exp\left[1/\sigma\right]}_{b'}\right) \tag{C.5}$$

$$\text{(replace with cos)} \quad = \alpha' \times \mathbf{w}^T\mathbf{x} \times \left(\exp\left[c(\mathbf{x},\mathbf{w})/\sigma\right] - b'\right) \tag{C.6}$$

Here, $c(\mathbf{x},\mathbf{w}) = \cos(\angle(\mathbf{x},\mathbf{w}))$ computes the cosine similarity between the vectors $\mathbf{w}$ and $\mathbf{x}$ and we have marked changes in the equation in red for clarity. If we now replace the exponential function by its first-order Taylor expansion ($\exp(s) = 1 + s + O(s^2)$), we obtain

$$h(\mathbf{x};\mathbf{w}) \approx \alpha' \times \mathbf{w}^T\mathbf{x} \times \left(1 + c(\mathbf{x},\mathbf{w})/\sigma - b'\right) \tag{C.7}$$

Finally, setting $b' = (\alpha'/\sigma) = 1$, we obtain function that looks very similar to the B-cos transformation with $\mathrm{B} = 2$ (cf. Equation (C.16) below):

$$h(\mathbf{x};\mathbf{w}) = \mathbf{w}^T\mathbf{x} \times c(\mathbf{x},\mathbf{w}) = \operatorname{sgn}\left(c(\mathbf{x},\mathbf{w})\right) \times \text{B-cos}(\mathbf{x};\mathbf{w},\mathrm{B}=2) \tag{C.8}$$

I.e., up to an additional sign factor of the cosine term, the B-cos transformation with $\mathrm{B} = 2$ can be seen as an approximation (see Equation (C.7)) of an RBF-gated linear transformation (see Equation (C.2)).

Importantly, this means that Deep RBF models[2] (RBF-gating in every single layer, as opposed to just at the output as in [ZHS18]) can indeed be trained to yield competitive performance on complex datasets such as ImageNet. To the best of our knowledge, the B-cos Networks are the first RBF-like models to achieve this. In future work, we plan to further analyse this relation and the implications that it carries.

[2]Note that as long as σ is chosen such that the first-order Taylor expansion is a good approximation, the B-cos Networks essentially implement equation Equation (C.2) up to the difference in the sign of the cosine term.

C.4.2 The B-cos Transformation as a Scaled Linear Transformation

In the following, we provide additional details on how to express the B-cos transformation as a scaled linear transformation and in matrix form.

As described in Equation (6.3) in Chapter 6, the B-cos transformation is given by

$$\text{B-cos}(\mathbf{x};\mathbf{w}) = \|\widehat{\mathbf{w}}\|\,\|\mathbf{x}\| \times |c(\mathbf{x},\widehat{\mathbf{w}})|^{\text{B}} \times \text{sgn}(c(\mathbf{x},\widehat{\mathbf{w}}))\;, \tag{C.9}$$

$$\text{with}\quad c(\mathbf{x},\mathbf{w}) = \cos(\angle(\mathbf{x},\mathbf{w}))\;, \tag{C.10}$$

$$\widehat{\mathbf{w}} = \mathbf{w}/\|\mathbf{w}\|\;, \tag{C.11}$$

$\angle(\mathbf{x},\mathbf{w})$ returning the angle between $\mathbf{x}$ and $\mathbf{w}$, and sgn the sign function. Note that the sign function can be expressed as $\text{sgn}(a)=a/|a|$ for $|a|\neq 0$ and zero otherwise. Hence, Equation (C.9) can be expressed as

$$\text{B-cos}(\mathbf{x};\mathbf{w}) = \|\widehat{\mathbf{w}}\|\,\|\mathbf{x}\| \times |c(\mathbf{x},\widehat{\mathbf{w}})|^{\text{B}} \times \text{sgn}(c(\mathbf{x},\widehat{\mathbf{w}})) \tag{C.12}$$

$$\text{(replace sgn)}\qquad = \|\widehat{\mathbf{w}}\|\,\|\mathbf{x}\| \times |c(\mathbf{x},\widehat{\mathbf{w}})|^{\text{B}} \times c(\mathbf{x},\widehat{\mathbf{w}})/|c(\mathbf{x},\widehat{\mathbf{w}})| \tag{C.13}$$

$$\text{(combine cos terms)}\qquad = \|\widehat{\mathbf{w}}\|\,\|\mathbf{x}\| \times |c(\mathbf{x},\widehat{\mathbf{w}})|^{\text{B}-1} \times c(\mathbf{x},\widehat{\mathbf{w}}) \tag{C.14}$$

$$\text{(reorder)}\qquad = \|\widehat{\mathbf{w}}\|\,\|\mathbf{x}\| \times c(\mathbf{x},\widehat{\mathbf{w}}) \times |c(\mathbf{x},\widehat{\mathbf{w}})|^{\text{B}-1} \tag{C.15}$$

$$\text{(write first three factors as linear transform)}\qquad = \widehat{\mathbf{w}}^T\mathbf{x} \times |c(\mathbf{x},\widehat{\mathbf{w}})|^{\text{B}-1}\;. \tag{C.16}$$

For clarity, we marked the changes between lines in the above equations in red.

From Equation (C.16) it becomes clear that a B-cos transformation simply computes a rescaled linear transform, as in Equation (6.4) in Chapter 6. Thus, multiple units in parallel (i.e., a layer $\mathbf{l}^*$ of B-cos units) can easily be expressed in matrix form via

$$\mathbf{l}^*(\mathbf{x}) = |c(\mathbf{x},\widehat{\mathbf{W}})|^{\text{B}-1} \times \widehat{\mathbf{W}}\mathbf{x}\;. \tag{C.17}$$

Here, the $\times$, cos, and absolute value operators are applied element-wise and the rows of $\widehat{\mathbf{W}}$ are given by $\widehat{\mathbf{w}}_n$ of the individual units n. Hence, the output of each unit (entry in output vector $\mathbf{l}^*$) is the down-scaled linear transformation from Equation (C.16). Note that Equation (C.17) is the same as Equation (6.10).

C.4.3 On the Relevance of Image Encoding for the Visualisations

We encode pixels as $[r,g,b,1-r,1-g,1-b]$, see Chapter 6. This has two important advantages.

On the one hand, as argued by [BFS21], this overcomes a bias towards bright pixels. For this, note that the model output is computed as a linear transformation of the input $\mathbf{x}$. As such, the contribution to the output per pixel is given by the weighted input strength. In particular, a specific pixel location (i,j) with colour channels c contributes $\sum_c w_{(i,j,c)}x_{(i,j,c)}$ to the output. Under the conventional encoding—i.e., $[r,g,b]$—, a black pixel is encoded by $x_{(i,j,c)}=0$ for $c\in\{1,2,3\}$ and can therefore not contribute to the model output. Since we train the model to maximise its outputs (binary cross entropy loss, see Section 6.1.2.2 in Chapter 6), the network will preferentially encode bright pixels, as these can produce higher contributions for maximising the output than dark pixels. In contrast, under the new encoding dark and bright pixels have the same amount of 'signal' that can be weighted, i.e., $\sum_c x_{(i,j,c)}=3\;\forall\;(i,j)$.

Moreover, this encoding allows to unambiguously infer the colour of a pixel solely based on the angle of the pixel vector $[r,g,b,1-r,1-g,1-b]$. To contrast this with the original encoding, consider a pixel that is (almost) completely black and given by $[r,g,b]$ with $g=0,b=0,r=0.001$. This pixel has the same angle as a red pixel, given by $r=1,g=0,b=0$. Thus, these two colors cannot be disambiguated

based on their angle. By adding the three additional colour channels $[1-r, 1-g, 1-b]$, each colour is uniquely encoded by the direction of the pixel vector. Finally, note that the B-cos transformation induces an alignment pressure on the weights, i.e., the model weights are optimised such that $\mathbf{W}_{1\to L}$ points in the same direction as (important features in) the input. Consequently, the weights will reproduce the *angles* of the pixels, but there is no constraint on their *norm*. Since the angle is sufficient for inferring the colour, we can nevertheless decode the angles of the weight vectors into RGB colors.

APPENDIX — MODEL GUIDANCE

D

In this supplement to our work on using explanations to guide models, we provide:

D.1 Additional Qualitative Results (VOC and COCO)

D.1.1 Qualitative Examples Across Losses, Attribution Methods, and Layers

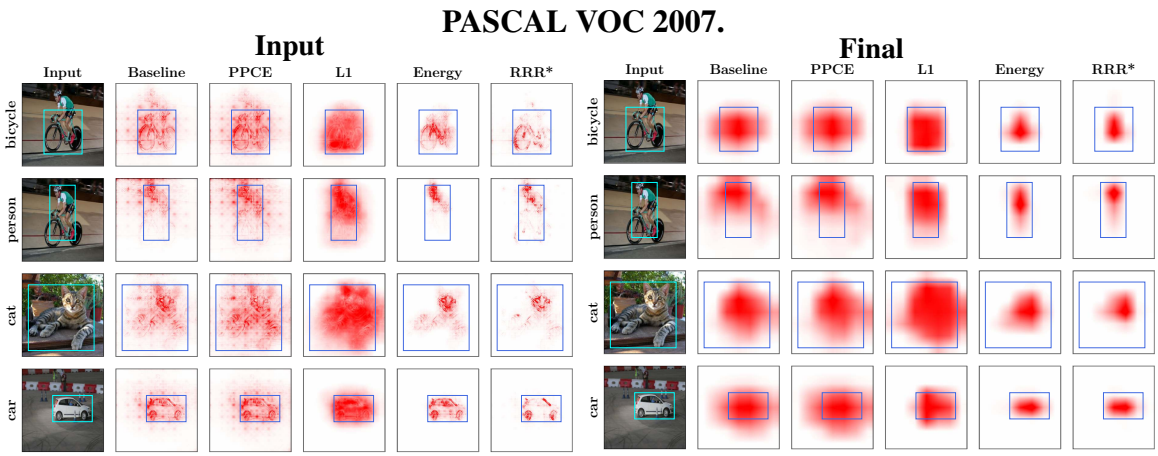
In Figures D.1 and D.2, we visualise attributions across losses, attribution methods, and layers for the same set of examples from the VOC and COCO datasets respectively. As discussed in Chapter 7, we make the following observations.

First, when guiding models at the *final layer*, we observe a marked improvement in the granularity of the attribution maps for all losses (**R5**), except for PPCE, for which we do not observe notable differences. The improvements are particularly noticeable on the COCO dataset (Figure D.2, "Final" column), in which the objects tend to be smaller. E.g., when looking at the airplane image (last row per model), we observe much fewer attributions in the background after applying model guidance.

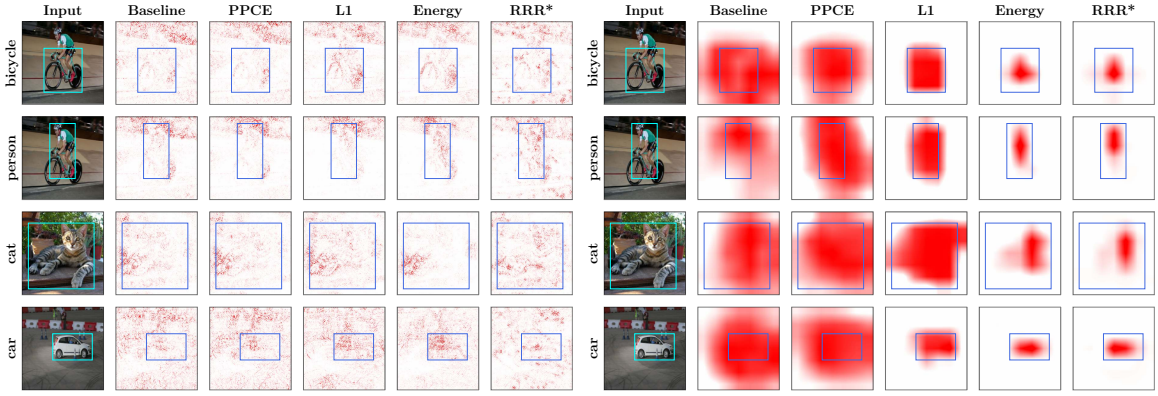
Second, as the L_1 loss optimises for uniform coverage *within* the bounding boxes, it provides coarser attributions that tend to fill the entire bounding box (cf. **R3**). This can be observed particularly well for the large objects from the VOC dataset: e.g., whereas models trained with the Energy and the RRR loss highlight just a relatively small area within the bounding box of the cat (Figure D.1, "Final" column, third row), the L_1 loss yields much more distributed attributions for all models.

Third, at the input layer, the B-cos models show the most notable qualitative improvements (cf. **R4**). In particular, although the $\mathcal{X}$-DNN models show some reduction in noisy background attributions (e.g. last rows in Figure D.1c and Figure D.2c), the attributions remain rather noisy for many of the images; for the Vanilla models, the improvements are even less pronounced (Figure D.1b, Figure D.2b). The B-cos models, on the other hand, seem to lend themselves better to such guidance being applied to the attributions at the input layer (Figure D.1a, Figure D.2a) and the resulting attributions show much more detail (Energy + RRR) or an increased focus on the entire bounding box (L_1). Especially with the Energy, the B-cos models are able to clearly focus on even small objects, see Figure D.2a.

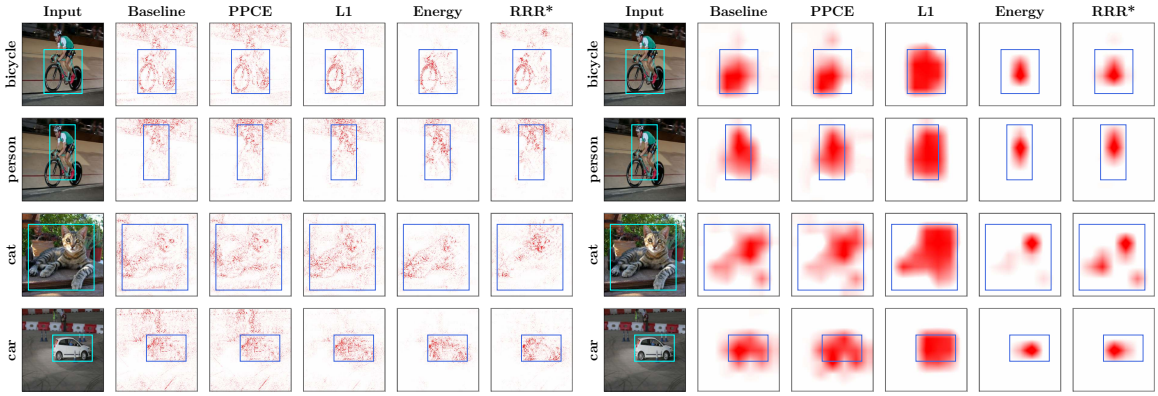
For additional results from both the VOC as well as the COCO dataset, please see Figure D.3.



(a) **B-cos ResNet-50**.

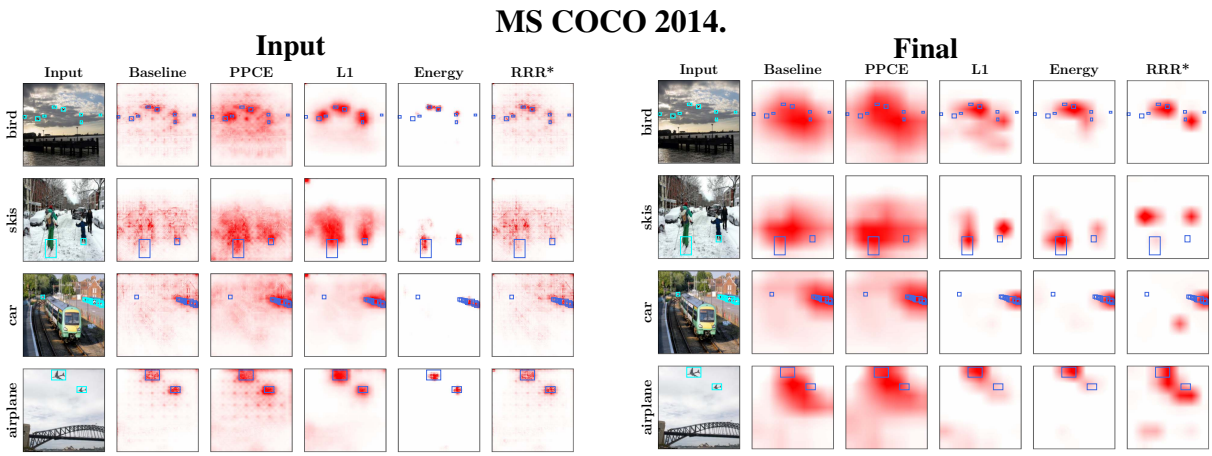


(b) **Vanilla ResNet-50**.

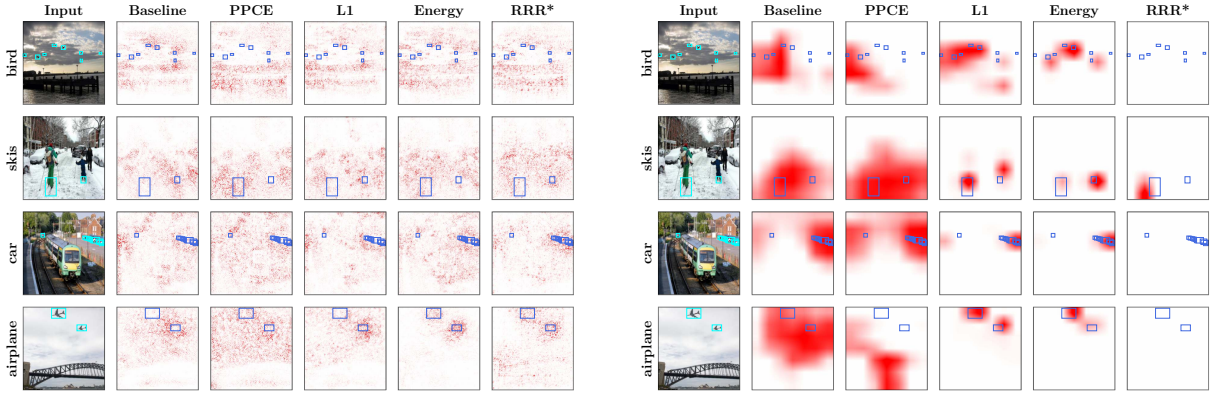


(c) $\mathcal{X}$**-DNN ResNet-50**.

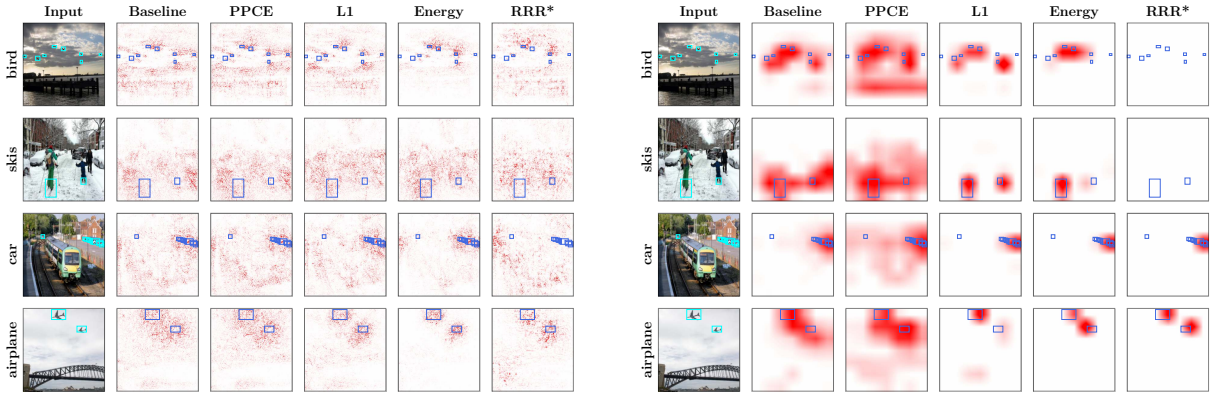
Figure D.1: **Qualitative examples from the VOC dataset**. This figure allows to compare between models (*major rows*, i.e. **(a), (b), and (c)**) losses (*major columns*) and layers (*left+right*) for multiple images (*minor rows*).



(a) **B-cos ResNet-50**.



(b) **Vanilla ResNet-50**.



(c) $\mathcal{X}$**-DNN ResNet-50**.

Figure D.2: **Qualitative examples from the COCO dataset**. This figure allows to compare between models (*major rows*, i.e. **(a), (b), and (c)**) losses (*major columns*) and layers (*left+right*) for multiple images (*minor rows*).

Figure D.3: **Qualitative examples from the VOC (*left*) and COCO (*right*) datasets**. In particular, here we just show additional examples for the B-cos models with input attributions, as this configuration exhibits the most detail. We show results for such models trained with different losses (*columns*) for multiple images (*rows*).

D.1.2 Additional Visualisations for Training with Coarse Bounding Boxes

In this section, we show more detailed and additional examples of models trained with coarser bounding boxes, i.e. with bounding boxes that are purposefully dilated during training by various amounts (10%, 25%, or 50%), see Figure D.4. In accordance with our findings in Chapter 7 (cf. **R8**), we observe that the Energy loss is highly robust to such 'annotation errors': the attribution maps improve noticeably in all cases (compare the Energy row with the respective baseline result). In contrast, the L_1 loss seems more dependent on high-quality annotations, which we also observe quantitatively, see Figure D.12.

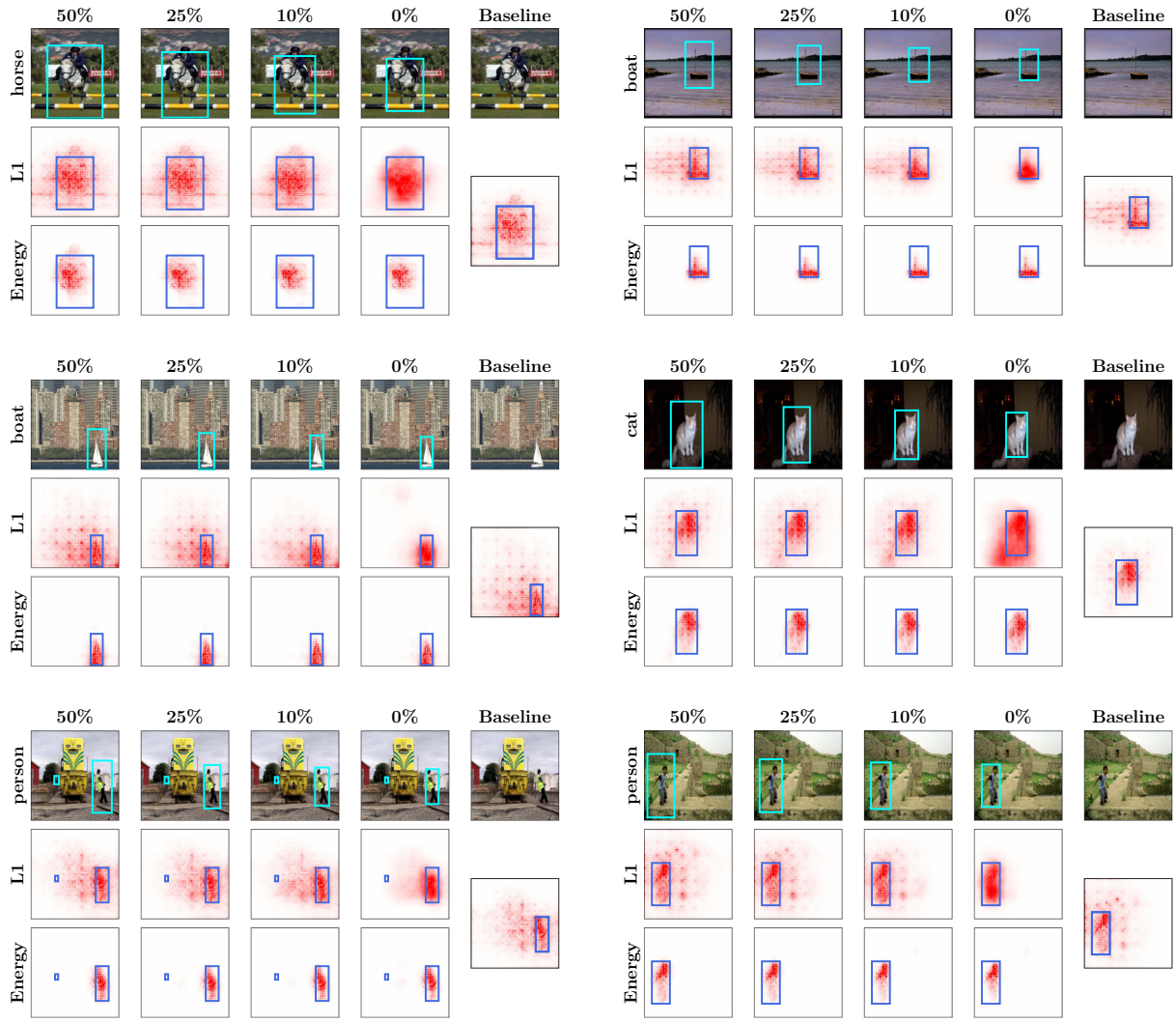


Figure D.4: **Qualitative examples of the impact of using coarse bounding boxes for guidance.** We show examples of B-cos attributions from the input layer on the baseline model and on models guided with the Energy and L_1 localisation losses with varying degrees of dilation $\{10\%, 25\%, 50\%\}$ in bounding boxes during training. For each example (**block** in the figure), we show the image and bounding boxes with varying degrees of dilation (**top** row), attributions with the L_1 localisation loss (**middle** row), and attributions with the Energy localisation loss (**bottom** row). We find that in contrast to using the L_1 localisation loss, guidance with Energy localisation loss maintains localisation of attributions to on-object features even with dilated bounding boxes. Note that bounding boxes are dilated only during training, not during evaluation. Bounding boxes in **light blue** show the extent of dilation that *would have been used* had the image been from the training set, while those in **dark blue** show undilated bounding boxes that are used during evaluation.

D.2 Additional Quantitative Results (VOC and COCO)

In this section, we provide additional quantitative results from our experiments on the VOC and COCO datasets. Specifically, in Section D.2.1, we show additional results comparing classification and localisation performance. In Section D.2.2 we present results for guiding models via GradCAM attributions. In Section D.2.3, we show that training at intermediate layers can be a cost-effective way approach to performing model guidance. In Section D.2.4, we evaluate how well the attributions localise to on-object features (as opposed to background features) within the bounding boxes, and find that the Energy outperforms other localisation losses in this regard. In Section D.2.5, we provide additional analyses regarding training with a limited number of annotated images. Finally, in Section D.2.6, we provide additional analyses regarding the usage of coarse, dilated bounding boxes during training.

D.2.1 Comparing Classification and Localisation Performance

In this section, we discuss additional quantitative findings with respect to localisation and classification performance metrics (IoU, mAP) for a selected subset of the experiments; for a full comparison of all layers and metrics, please see Figures D.15 to D.18.

Additional IoU Results. In Figures D.5 and D.6, we show the remaining results comparing IoU vs. F1 scores that were not shown in Chapter 7 for VOC and COCO respectively. Similar to the results in Chapter 7 for the EPG metric (Figure 7.5), we find that the results between datasets are highly consistent for the IoU metric.

In particular, as discussed in Section 7.3.1, we find that the L_1 loss yields the largest improvements in IoU when optimised at the final layer, see bottom rows of Figures D.5 and D.6. At the input layer, we find that Vanilla and $\mathcal{X}$-DNN ResNet-50 models are not improving their IoU scores noticeably, whereas the B-cos models show significant improvements. We attribute this to the noisy patterns in the attribution maps of Vanilla and $\mathcal{X}$-DNN models, which might be difficult to optimise.

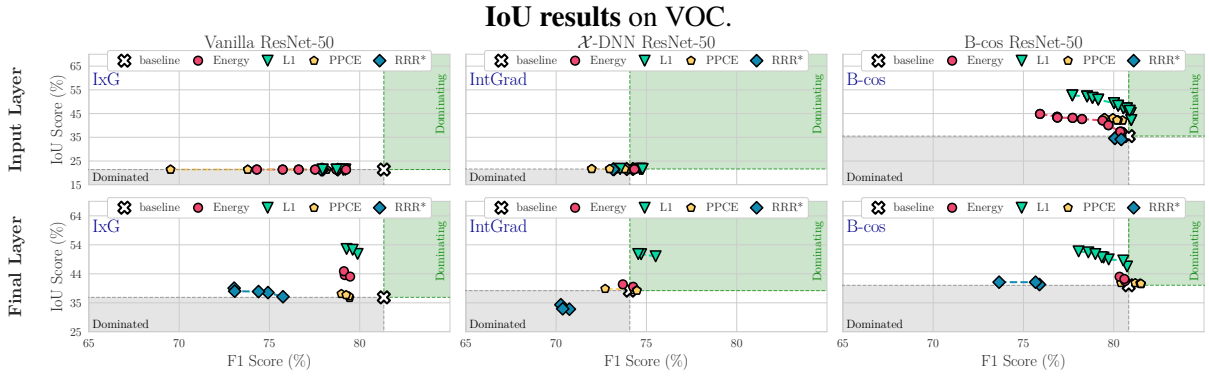


Figure D.5: **IoU results on PASCAL VOC 2007.** We show IoU vs. F1 for all localisation loss functions, attribution methods, and layers. In contrast to the consistent improvements observed at the final layer with the L_1 loss, the IoU metric only noticeably improves for the B-cos models after model guidance. We attribute this to the high amount of noise present in the attribution maps of Vanilla and $\mathcal{X}$-DNN models, see e.g. Figures D.1 and D.2. For results on the COCO dataset, please see Figure D.6.

Using mAP to Evaluate Classification Performance. In all results so far, we plotted the localisation metrics (EPG, IoU) versus the F1 score as a measure of classification performance. In order to highlight

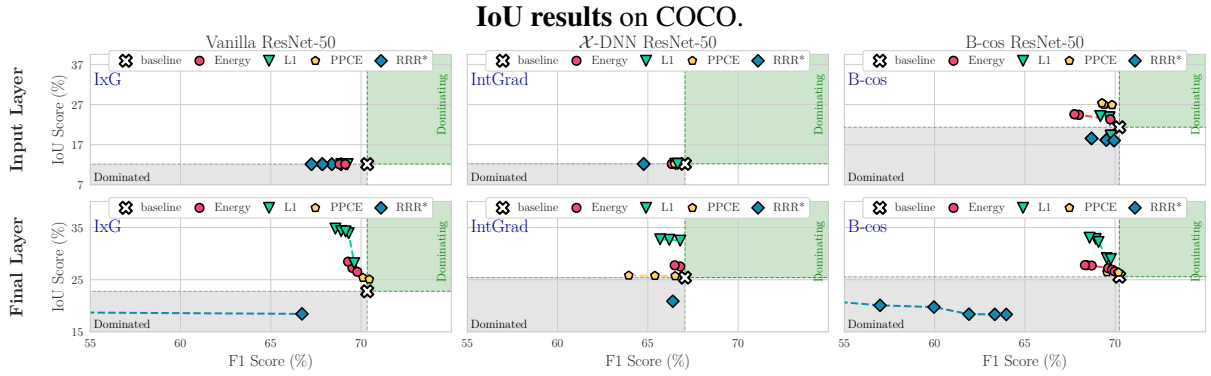


Figure D.6: **IoU results on MS COCO 2014.** We show IoU vs. F1 for all localisation loss functions, attribution methods, and layers. In contrast to the consistent improvements observed at the final layer with the L_1 loss, the IoU metric only noticeably improves for the B-cos models after model guidance. We attribute this to the high amount of noise present in the attribution maps of Vanilla and $\mathcal{X}$-DNN models, see e.g. Figures D.1 and D.2. For results on the VOC dataset, please see Figure D.5.

that the observed trends are independent of this particular choice of metric, in Figure D.7, we show both EPG as well as IoU results plotted against the mAP score.

In general, we find the results obtained for the mAP metric to be highly consistent with the previously shown results for the F1 metric. E.g., across all configurations, we find the Energy to yield the highest gains in EPG score, whereas the L_1 loss provides the best trade-offs with respect to the IoU metric. In order to easily compare between all results for all datasets and metrics, please see Figures D.15 to D.18.

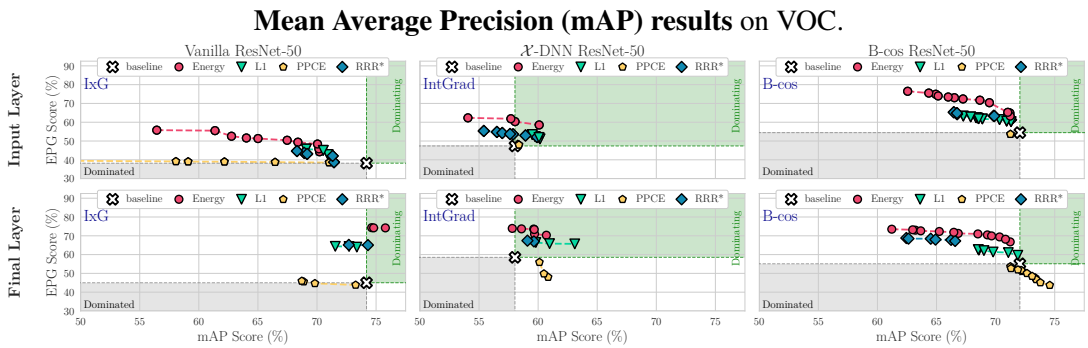
D.2.2 Model Guidance via GradCAM

In Figure D.8, we show the EPG vs. F1 results of training models with GradCAM applied at the final layer on the VOC dataset; for IoU results and results on COCO, please see Figures D.19 and D.20. When comparing between rows (**top:** results of Chapter 7; **bottom:** GradCAM), it becomes clear that GradCAM performs very similarly to IxG / IntGrad / B-cos attributions on Vanilla / $\mathcal{X}$-DNN / B-cos models. In fact, note that GradCAM is very similar to IxG and IntGrad (equivalent up to an additional zero-clamping) for the respective models and any differences in the results can be attributed to the non-deterministic training pipeline and the similarity between the results should thus be expected.

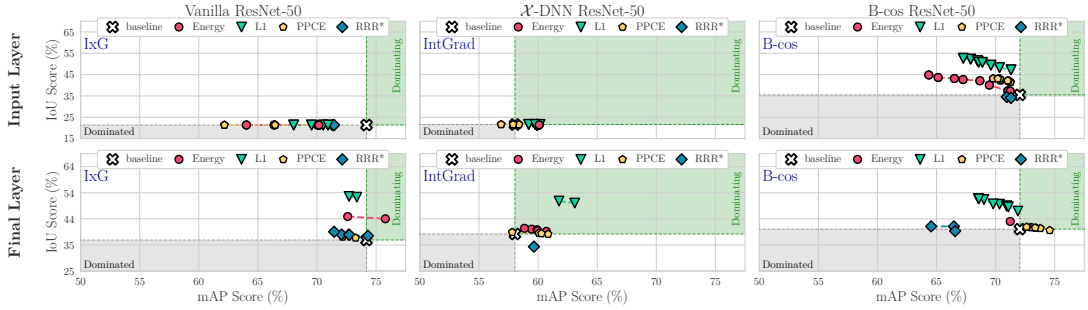
D.2.3 Model Guidance at Intermediate Layers

In Section 7.3, we show results for guidance on two 'model depths', i.e. at the input and the final layer. This corresponds to the two depths at which attributions are typically computed, e.g. IxG and IntGrad are typically computed at the input, while GradCAM is typically computed using final spatial layer activations. Following [RBS22], for a fair comparison we optimise using each attribution methods at identical depths. For the final and intermediate layers in the network, this is done by treating the output activations at that layer as effective inputs over which attributions are to be computed. As done with GradCAM [SCD+17], we then upscale the attribution maps to image dimensions using bilinear interpolation and then use them for model guidance.

In Figure D.9, we show results for performing model guidance at additional intermediate layers: Mid1, Mid2, and Mid3. Specifically, for the ResNet-50 models we use, these layers correspond to the outputs of



(a) **EPG vs. mAP.**



(b) **IoU vs. mAP.**

Figure D.7: **Quantitative comparison of EPG and IoU vs. mAP scores for VOC.** To ensure that the trends observed in Chapter 7 generalise beyond the F1 metric, in this figure we show the EPG and IoU scores plotted against the mAP metric. We find the results obtained for the mAP metric to be highly consistent with the previously shown results for the F1 metric, see e.g. Figures 7.5 and 7.6. E.g., across all configurations, we find the Energy to yield the highest gains in EPG score, whereas the L_1 loss provides the best trade-offs with respect to the IoU metric. To compare between all results for all datasets and metrics, please see Figures D.7, D.15, D.16 and D.18.

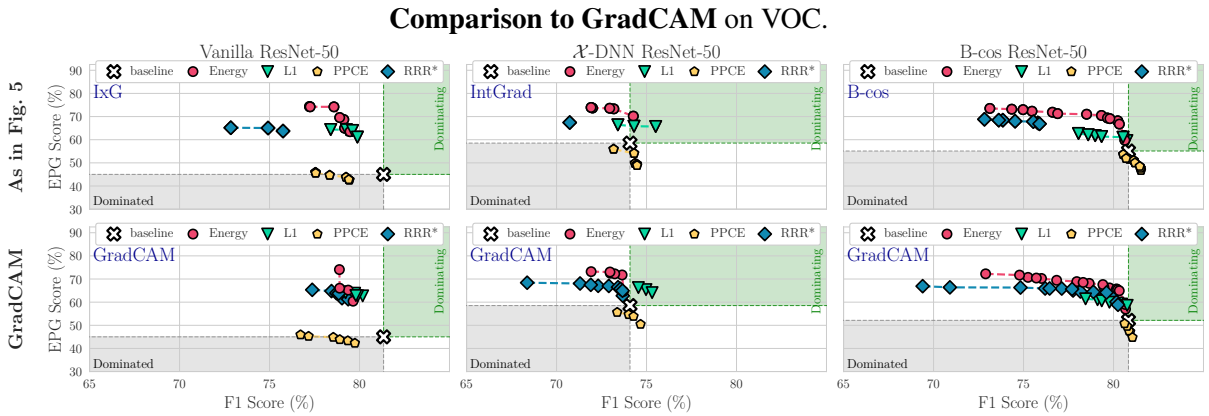


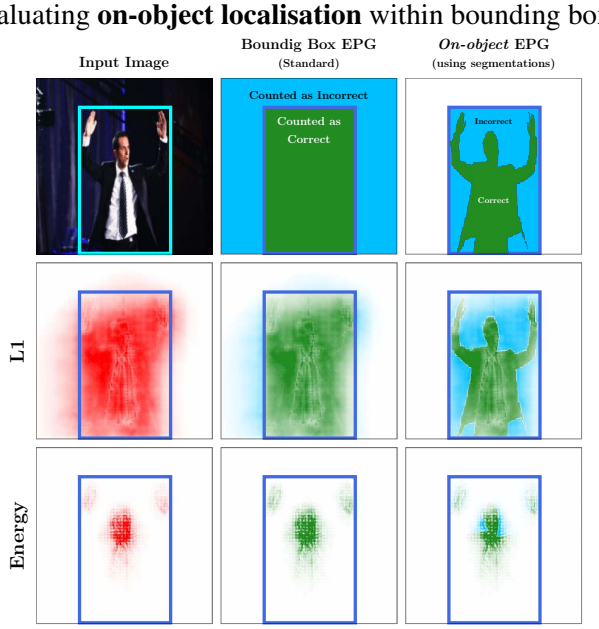
Figure D.8: **Quantitative results using GradCAM.** We show EPG scores vs. F1 scores for all localisation losses and models using GradCAM at the final layer (**bottom row**) and compare it to the results shown in Chapter 7 (**top row**). As expected, GradCAM performs very similarly to IxG (Vanilla) and IntGrad ($\mathcal{X}$-DNN) used at the final layer—in particular, note that for ResNet-50 architectures, IxG and IntGrad are very similar to GradCAM for Vanilla and $\mathcal{X}$-DNN models respectively (see Section D.2.2). Similarly, we find GradCAM to also perform comparably to the B-cos explanations when used at the final layer; IoU results and results on COCO, in Figures D.19 and D.20.

`conv2_x`, `conv3_x`, and `conv4_x` respectively in the ResNet nomenclature ([HZRS16]), while the final layer corresponds to the output of `conv5_x`. We find that the EPG performance at these intermediate layers through the network follows the trends when moving from the input to the final layer. Similar results for IoU can be found in Figure D.22.

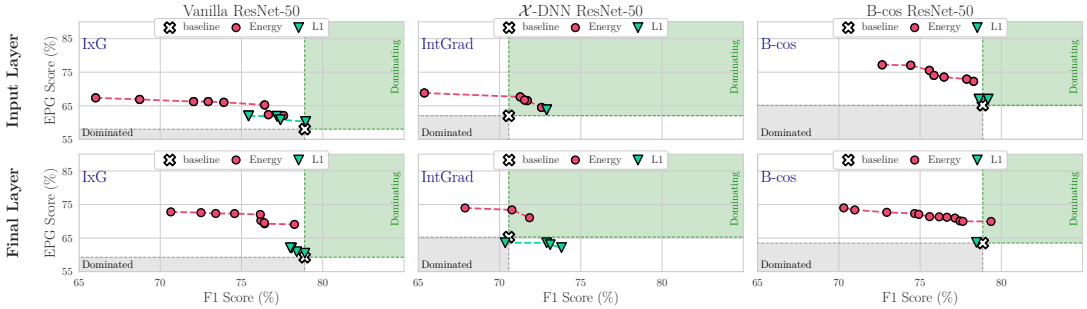
Figure D.9: **Intermediate layer results comparing EPG vs. F1.** We compare the effectiveness of model guidance at varying network depths (**rows**) for each attribution method and model (**columns**) across localisation loss functions. For the B-cos model, we find similar trends at all network depths, with the Energy localisation loss outperforming all other losses. For the Vanilla and $\mathcal{X}$-DNN models, the Energy loss similarly performs the best, but we also observe improved performance across losses when optimising at deeper layers of the network. Full results can be found in Figures D.21 and D.22.

D.2.4 Evaluating On-Object Localisation

The standard EPG metric (Equation (7.2)) evaluates the extent to which attributions localise to the bounding boxes. However, since such boxes often include background regions, the EPG score does not distinguish between attributions that focus on the object and attributions that focus on such background regions within the bounding boxes.



(a) **Evaluating *on-object* localisation within the bounding boxes: On-object EPG.** In the standard EPG metric (**middle** column), we compute the fraction of positive attributions within the bounding boxes. In other words, attributions within the bounding box (**green** region) positively impact the metric, while attributions outside (**blue** region) negatively impact it. Since bounding boxes are coarse annotations and often include background regions, the standard EPG does not evaluate how well attributions localise *on-object* features, e.g. the person in the figure. To measure this, we evaluate with an additional Segmentation EPG metric (**right** column), where we compute the fraction of positive attributions in the bounding box that lie within the segmentation mask of the object. Here, attributions within the segmentation mask (**green** region) positively impact the metric, and attributions outside the segmentation mask and inside the bounding box (**blue** region) negatively impact it. Note that attributions outside the bounding box have no effect on Segmentation EPG. As an example and to visualise qualitative differences between losses, in the bottom rows (L_1, Energy), we show attributions for a B-cos model guided at the input layer. As becomes clear, by employing a uniform prior on attributions within the bounding box, the L_1 loss is effectively optimised to fill the entire bounding box and thus to not only highlight *on-object* features. This can also be observed quantitatively, see e.g. Figure D.10b, right column.



(b) **On-object EPG results.** We evaluate across models (**columns**) and layers (**rows**) for the Energy and L_1 localisation losses. As seen qualitatively (e.g. Figure 7.9), we find that the Energy loss is more effective than the L_1 loss in localising attributions to the object as opposed to background regions within the bounding boxes. This is explained by the fact that the L_1 loss promotes uniformity in attributions within the bounding box, and treats both on-object and background features inside the box with equal importance, while the Energy loss only optimises for attributions to lie within the bounding box without placing any constraints on where they may lie, leaving the model free to decide which regions within the box are important for its decision.

Figure D.10: **Evaluating *on-object* localisation via EPG.** We show **(a)** the schema for the on-object EPG metric and how it differs from the bounding box EPG, and **(b)** quantitative results on evaluating with on-object EPG.

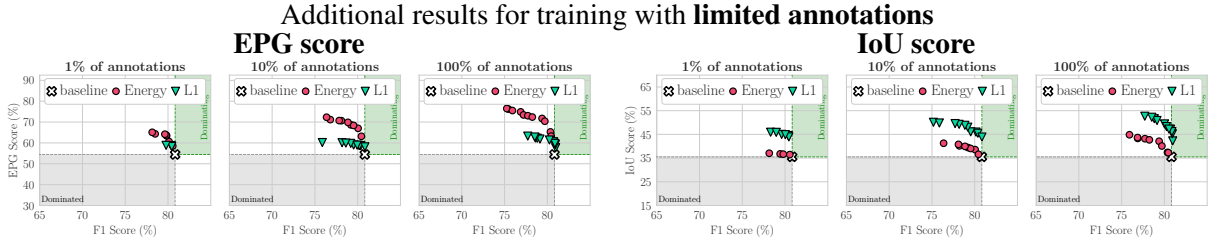


Figure D.11: **EPG and IoU scores for limited annotations.** We show EPG vs. F1 (**left**) and IoU vs. F1 (**right**) for B-cos attributions at the input when optimising with the Energy and L_1 localisation losses, when using $\{1\%, 10\%, 100\%\}$ training annotations. We find that model guidance is generally effective even when training with annotations for a limited number of images. While the performance slightly worsens when using 1% annotations, using just 10% annotated images yields similar gains to using a fully annotated training set. Full results can be found in Figures D.23 and D.24.

To additionally evaluate for on-object localisation, we use a variant of EPG that we call On-object EPG. In contrast to standard EPG, we compute the fraction of positive attributions in pixels contained within the segmentation mask of the object out of positive attributions within the bounding box. This measures how well attributions *within the bounding boxes* localise to the object, and is not influenced by attributions outside the bounding boxes. A visual comparison of the two metrics is shown in Figure D.10.

We find that the Energy localisation loss outperforms the L_1 localisation loss both qualitatively (Figure D.10a) and quantitatively (Figure D.10b) on this metric. This is explained by the fact that the L_1 promotes uniformity in attributions across the bounding box, giving equal importance to on-object and background features within the box. In contrast, the Energy loss only optimises for attributions to lie within the box, without any constraint on *where* in the box they lie. This also corroborates our previous qualitative observations (e.g. Figure 7.9).

D.2.5 Model Guidance with Limited Annotations

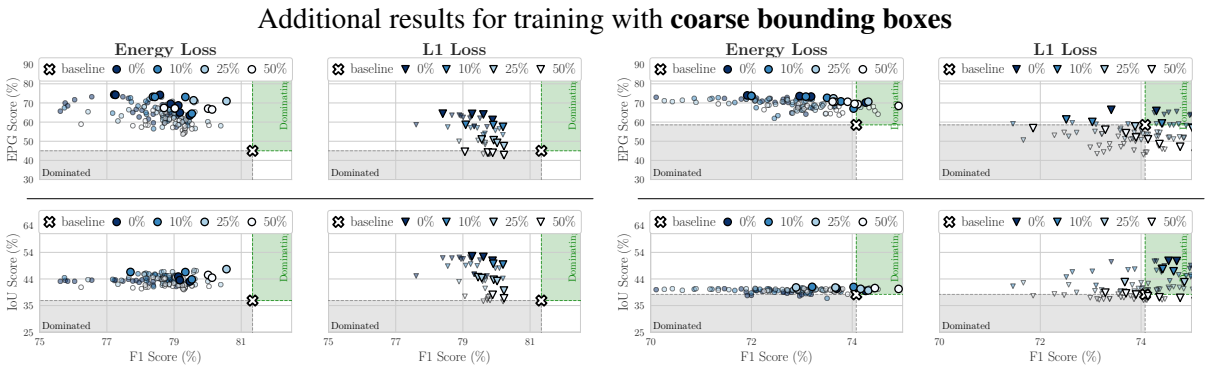
In Figure D.11, we show the impact of using limited annotations when training (Section 7.3.4) when optimising with the Energy and L_1 localisation losses for B-cos attributions at the input. We find that in addition to EPG, trends in IoU scores also remain consistent even when using bounding boxes for just 1% of the the training images.

D.2.6 Model Guidance with Noisy Annotations

In Figure D.12, we additionally show the impact of training with coarse, dilated bounding boxes for IxG attributions on the Vanilla model, and IntGrad attributions on the $\mathcal{X}$-DNN model. Similar to the results seen with B-cos attributions (Figure 7.10), we find that the Energy localisation loss is robust to coarse annotations, while the performance with L_1 localisation loss worsens as the dilations increase.

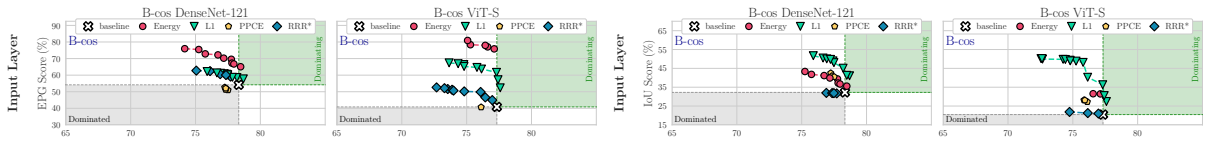
D.2.7 Evaluation on DenseNet and ViT Models

In Figure D.13, we evaluate the best performing configurations from our study, i.e. performing guidance using B-cos attributions at input, on additional model backbones, specifically DenseNet-121 and ViT-S. We find that the trends observed with ResNet-50 models generalises to these backbones, with the Energy loss yielding the highest gains for EPG, and the L_1 loss yielding the highest gains for IoU.



(a) **Vanilla ResNet-50 @ Final.** (b) **$\mathcal{X}$-DNN ResNet-50 @ Final.**

Figure D.12: **Coarse bounding box results.** We show the impact of dilating bounding boxes during training for the **(a)** Vanilla and **(b)** $\mathcal{X}$-DNN models. Similar to the results seen with B-cos models (Figure 7.10), we find that the Energy localisation loss is generally robust to coarse annotations, while the effectiveness of guidance with the L_1 localisation loss worsens as the extent of coarseness (dilations) increases. Full results in Figure D.25.



(a) (b)

Figure D.13: **EPG and IoU vs. F1 on VOC for two additional B-cos architectures.** We find that the trends observed in Chapter 7 for a B-cos ResNet-50 backbone (cf. Figures 7.5 and 7.6, right columns) generalise to other backbone architectures. In particular, we find that the L_1 loss yields the highest gains in IoU, whereas the Energy loss yields the highest gains in EPG, both for a DenseNet-121 and a ViT-S model.

D.3 WATERBIRDS RESULTS

			Conventional Setting					Reversed Setting				
	Layer	Loss	G1 Acc	G2 Acc	G3 Acc	G4 Acc	Overall	G1 Acc	G2 Acc	G3 Acc	G4 Acc	Overall
B-cos	Inp.	Energy	99.2 (±0.1)	40.4 (±1.0)	**56.1 (±4.0)**	96.6 (±0.4)	**71.2 (±0.1)**	**99.4 (±0.1)**	**70.2 (±2.1)**	**62.8 (±2.1)**	96.5 (±0.6)	**83.6 (±1.1)**
		L1	99.3 (±0.1)	37.0 (±0.8)	51.1 (±1.9)	**97.2 (±0.3)**	69.5 (±0.2)	99.3 (±0.3)	67.7 (±3.3)	58.8 (±5.0)	**96.7 (±0.7)**	82.2 (±0.9)
	Fin.	Energy	99.3 (±0.1)	**41.0 (±2.1)**	53.1 (±0.8)	96.3 (±0.5)	71.1 (±0.9)	**99.4 (±0.2)**	70.1 (±3.1)	60.2 (±3.9)	95.8 (±1.1)	83.2 (±1.1)
		L1	99.3 (±0.1)	34.3 (±3.2)	49.4 (±2.6)	96.6 (±0.6)	68.2 (±1.1)	**99.4 (±0.1)**	69.8 (±2.1)	56.3 (±1.8)	96.1 (±0.7)	82.8 (±0.8)
	Baseline		**99.4 (±0.1)**	37.2 (±0.2)	43.4 (±2.4)	96.5 (±0.1)	68.7 (±0.2)	**99.4 (±0.1)**	62.8 (±0.2)	56.6 (±2.4)	96.5 (±0.1)	80.1 (±0.2)
$\mathcal{X}$-DNN	Inp.	Energy	99.3 (±0.2)	**47.0 (±9.1)**	49.2 (±4.8)	**96.8 (±0.7)**	**73.1 (±3.4)**	99.0 (±0.3)	**67.6 (±4.8)**	**63.9 (±3.6)**	96.1 (±0.7)	**82.6 (±2.0)**
		L1	99.1 (±0.6)	40.4 (±7.3)	41.8 (±3.8)	96.5 (±0.6)	69.6 (±3.2)	**99.3 (±0.2)**	59.1 (±4.7)	63.6 (±6.1)	96.0 (±0.9)	79.3 (±1.3)
	Fin.	Energy	99.2 (±0.4)	42.5 (±10.4)	**54.2 (±3.2)**	96.6 (±0.9)	71.9 (±4.2)	99.2 (±0.2)	65.3 (±2.0)	62.3 (±3.3)	96.0 (±0.5)	81.5 (±0.9)
		L1	**99.4 (±0.1)**	45.1 (±4.0)	42.8 (±2.8)	96.5 (±0.5)	71.7 (±1.4)	**99.3 (±0.2)**	62.9 (±4.8)	59.8 (±4.8)	95.8 (±0.7)	80.4 (±1.8)
	Baseline		99.3 (±0.1)	39.8 (±0.7)	38.6 (±2.5)	96.3 (±0.7)	69.1 (±0.6)	**99.3 (±0.1)**	60.2 (±0.7)	61.4 (±2.5)	**96.3 (±0.7)**	79.6 (±0.5)
Vanilla	Inp.	Energy	99.4 (±0.2)	42.4 (±2.6)	47.9 (±3.5)	97.1 (±0.4)	71.2 (±1.0)	**99.6 (±0.2)**	50.7 (±7.3)	52.4 (±1.7)	97.2 (±0.5)	75.1 (±2.9)
		L1	**99.5 (±0.1)**	46.1 (±4.4)	51.1 (±4.0)	97.5 (±0.1)	73.1 (±1.6)	**99.6 (±0.1)**	48.0 (±7.8)	49.7 (±3.7)	96.8 (±0.6)	73.7 (±2.7)
	Fin.	Energy	**99.5 (±0.0)**	56.1 (±7.0)	**60.7 (±5.5)**	97.0 (±0.5)	**78.1 (±2.6)**	99.5 (±0.1)	59.4 (±5.9)	**56.5 (±3.7)**	97.2 (±0.5)	**78.9 (±1.9)**
		L1	**99.5 (±0.1)**	**57.1 (±2.9)**	55.4 (±2.5)	96.7 (±0.6)	77.8 (±1.0)	99.5 (±0.1)	56.3 (±6.7)	51.6 (±3.1)	97.3 (±0.6)	77.1 (±2.5)
	Baseline		99.4 (±0.0)	39.6 (±0.7)	53.7 (±2.1)	**97.7 (±0.0)**	70.8 (±0.0)	99.4 (±0.0)	**60.4 (±0.7)**	46.3 (±2.1)	**97.7 (±0.0)**	78.1 (±0.1)

Table D.1: **Classification performance on Waterbirds** after model guidance with the L_1 and the Energy loss. We find that both losses consistently improve the models' classification performance over the baseline model (i.e. a model without guidance). These improvements are particularly pronounced in the groups *not seen during training*, i.e. landbirds on water ("G2") and waterbirds on land ("G3"). For qualitative visualisations of the effect of model guidance on the waterbirds dataset, see Figure D.14.

As discussed in Section 7.3.5, we use the Waterbirds-100 dataset [SKHL20, PDN+22] to evaluate the effectiveness of model guidance in a setting where strong spurious correlations are present in the training data. This dataset consists of four groups—*Landbird* on *Land* (**G1**), *Landbird* on *Water* (**G2**), *Waterbird* on *Land* (**G3**), and *Waterbird* on *Water* (**G4**)—of which only groups **G1** and **G4** appear during training and the background is thus perfectly correlated with the type of bird (e.g. Landbird on land).

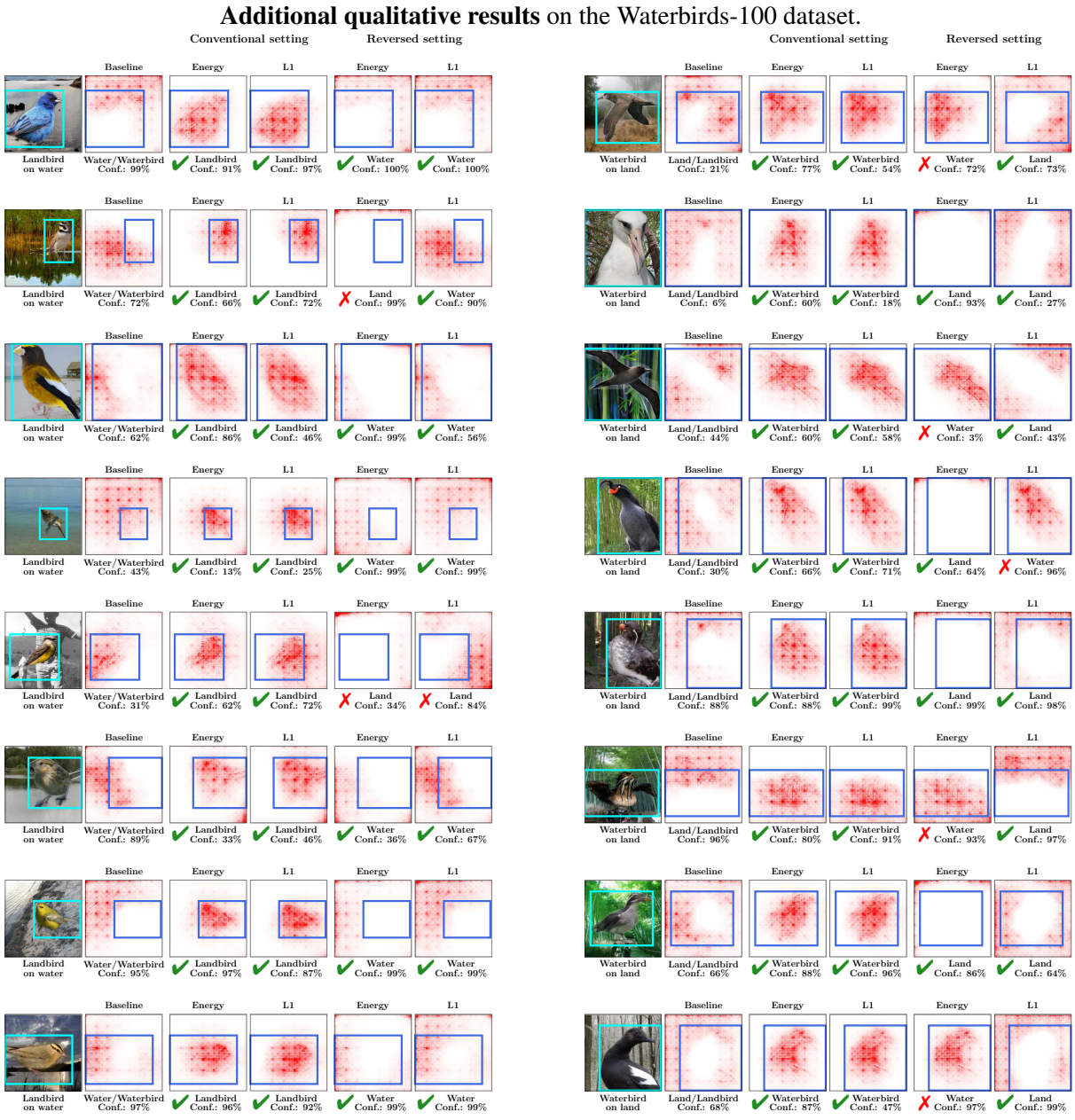
To evaluate the effectiveness of model guidance, we train the models on two binary classification tasks: to classify the type of birds (the *conventional setting*) or the background (the *reversed setting*, as described in [PDN+22]) and evaluate models without guidance (baselines), as well as with guidance: specifically, for guiding the models, we evaluate different models (Vanilla, $\mathcal{X}$-DNN, B-cos) with different guidance losses (Energy, L_1) applied at different layers (Input and Final), see Table D.1. For each model, we use its corresponding attribution method, i.e. IxG for Vanilla, IntGrad for $\mathcal{X}$-DNN, and B-cos for B-cos.

In Table D.1 we present the classification performance for the individual groups (**G1-G4**) as well as the average over all samples ('Overall') across all configurations; note that the group sizes differ in the test set and the average over the individual group acccuracies thus differs from the overall accuracy. For each row, the results are averaged over 4 runs (2 random training seeds and 2 different sets of 1% annotated samples) with the exception of the baseline results being an average over 2 runs.

In almost all cases, we find that both of the evaluated losses (Energy, L_1) improve the models' classification performance over the baseline. As expected, these improvements are particularly pronounced in the groups not seen during training, i.e. landbirds on water (**G2**) and waterbirds on land (**G3**).

Further, in Figure D.14, we show attribution maps of the baseline models, as well as the guided models.

As can be seen, model guidance not only improves the accuracy, but is also reflected in the attribution maps: e.g., in row 1 of Figure D.14a, we see that while the baseline model originally focused on the background (water) to classify the image, it is possible to guide the model to use the desired features (i.e. the bird in conventional setting and the background in the reversed setting) and consequently arrive at the desired classification decision. As this guidance is 'soft', we also observe cases in which the model still focused on the wrong feature and thus arrived at the wrong prediction: e.g. in Figure D.14b row 1 (reversed setting), the Energy-guided model still focuses on the bird and thus incorrectly predicts 'Water', similar to the L_1-guided model in row 4.



(a) **Landbirds on Water.** (b) **Waterbirds on Land.**

Figure D.14: **Qualitative results for the Waterbirds dataset.** Specifically, we show input layer attributions for B-cos models trained without guidance ('Baseline') as well as guided via the Energy or L_1 loss. We find that model guidance can be effective both for focusing on the bird and the background. For example, in the top row of (a), the model originally focuses on the background (col. 2) and classifies the image (col. 1) as Water/Waterbird. In the conventional setting, both the Energy and L_1 localisation losses are effective in redirecting the focus to the bird (cols. 3-4), changing the model's prediction to Landbird with high confidence. Similarly, in the reversed setting, both localisation losses direct the focus to the background (cols. 5-6), which increases the model's confidence in classifying the image as Water.

D.4 Implementation Details

D.4.1 Training and Evaluation Details

Implementations. We implement our code using PyTorch[1] [PGM+19]. The PASCAL VOC 2007 [EVGW+09] and MS COCO 2014 [LMB+14] datasets and the Vanilla ResNet-50 model were obtained from the Torchvision library[2] [PGM+19, mc16]. Official implementations were used for the B-cos[3] [BSFS24] and $\mathcal{X}$-DNN[4] [HSMR21] networks. Some of the utilities for data loading and evaluation were derived from NN-Explainer[5] [SPA+22], and for visualisation from the Captum library[6] [KMM+20].

D.4.1.1 *Experiments with VOC and COCO*

Training Baseline Models. We train starting from models pre-trained on ImageNet [DDS+09]. We fine-tune with fixed learning rates in $\{10^{-3}, 10^{-4}, 10^{-5}\}$ using an Adam optimiser [KB15] and select the checkpoint with the best validation F1-score. For VOC, we train for 300 epochs, and for COCO, we train for 60 epochs.

Training Guided Models. We train the models jointly optimised for classification and localisation (Equation (7.1)) by fine-tuning the baseline models. We use a fixed learning rate of 10^{-4} and a batch size of 64. For each configuration (given by a combination of attribution method, localisation loss, and layer), we train using three different values of λ_{loc}, as detailed in Table D.2. For VOC, we train for 50 epochs, and for COCO, we train for 10 epochs.

Selecting Models to Visualise. As described in Section 7.2, we select and evaluate on the set of Pareto-dominant models for each configuration after training. Each model on the Pareto front represents the extent of trade-off made between classification (F1) and localisation (EPG) performance. In practice, the 'best' model to choose would depend on the requirements of the end user. However, to evaluate the effectiveness of model guidance (e.g. Figures 7.1, 7.2 and 7.9), we select a representative model on the front whose attributions we visualise. This is done by selecting the model with the highest EPG score with an at most 5 p.p. drop in F1-score.

Efficient Optimisation. As described in Section 7.1.5, for each image in a batch, we optimise for localisation of a single class selected at random. This approximation allows us to perform model guidance efficiently and keeps the training cost tractable. However, to accurately evaluate the impact of this optimisation, we evaluate the localisation of all classes in the image at test time.

Training with Limited Annotations. As described in Section 7.3.4, we show that training with a limited number of annotations can be a cost effective way of performing model guidance. In order to maintain the relative magnitude of $\mathcal{L}_{\text{loc}}$ as compared to $\mathcal{L}_{\text{class}}$ in this setting, we scale up the values of λ_{loc} when training. The values of λ_{loc} we use are shown in Table D.3.

[1] https://github.com/pytorch/pytorch
[2] https://github.com/pytorch/vision
[3] https://github.com/B-cos/B-cos-v2
[4] https://github.com/visinf/fast-axiomatic-attribution
[5] https://github.com/stevenstalder/NN-Explainer
[6] https://github.com/pytorch/captum

Localisation Loss	**Values of λ_{loc}**
Energy	5×10^{-4}, 1×10^{-3}, 5×10^{-3}
L_1	1×10^{-3}, 5×10^{-3}, 1×10^{-2}
PPCE	1×10^{-4}, 5×10^{-4}, 1×10^{-3}
RRR*	5×10^{-6}, 1×10^{-5}, 5×10^{-5}

Table D.2: **Hyperparameter λ_{loc}: Default training.** used for when training on VOC and COCO with each localisation loss. Different values are used for different loss functions since the magnitudes of each loss varies.

Localisation Loss	**Values of λ_{loc}**
Energy	0.05, 0.100, 0.50
L_1	0.01, 0.100, 1.00

Table D.3: **Hyperparameter λ_{loc}: Limited annotations.** used for when training on VOC and COCO with **limited data** for each localisation loss. Different values are used for different loss functions since the magnitudes of each loss varies. We use larger values of λ_{loc} when training with limited annotations to maintain the relative magnitudes of the classification and localisation losses during training.

D.4.1.2 *Experiments with Waterbirds-100*

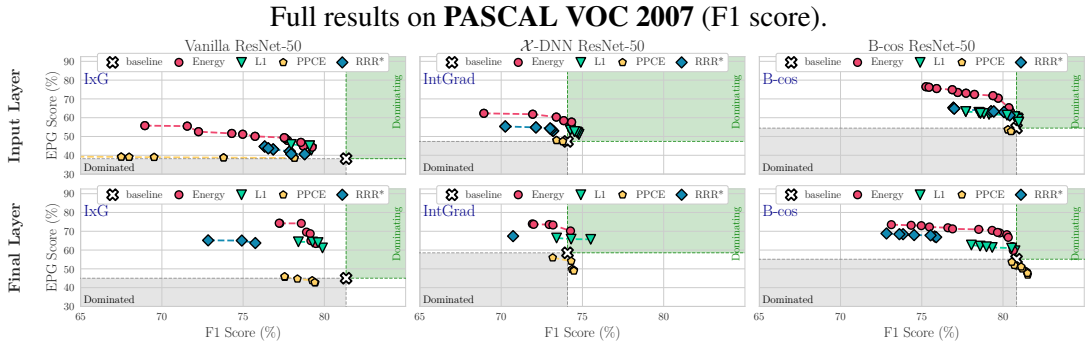
Data Distributions. The conventional binary classification task includes classifying *Landbird* from *Waterbird*, irrespective of their backgrounds. We use the same splits generated and published by [PDN+22]. As discussed in Section D.3, at training time there are no samples from **G2** or **G3**, making the bird type and the background perfectly correlated. In contrast, both the validation and test sets are balanced across foregrounds and backgrounds, i.e. a landbird is equally likely to occur on land or water, and vice-versa. However, as noted by [SKHL20], using a validation set with the same distribution as the test set leaks information on the test distribution in the process of hyperparameter and checkpoint selection during training. Therefore, we modify the validation split to avoid such information leakage; in particular, we use a validation set with the same distribution as the training set, and only use examples of groups **G1** and **G4**. Note that Table 7.1 refers to **G3** as the "Worst Group".

Training Details. We train starting from models pre-trained on ImageNet [DDS+09]. We fine-tune with fixed learning rate of 10^{-5} with λ_{loc} of 5×10^{-2} ($5 \times 10^{-4} \times 100$ for using 1% of annotations) using an Adam optimiser [KB15] . We train for 350 epochs with random cropping and horizontal flipping and select the checkpoint with the highest accuracy on the modified validation set.

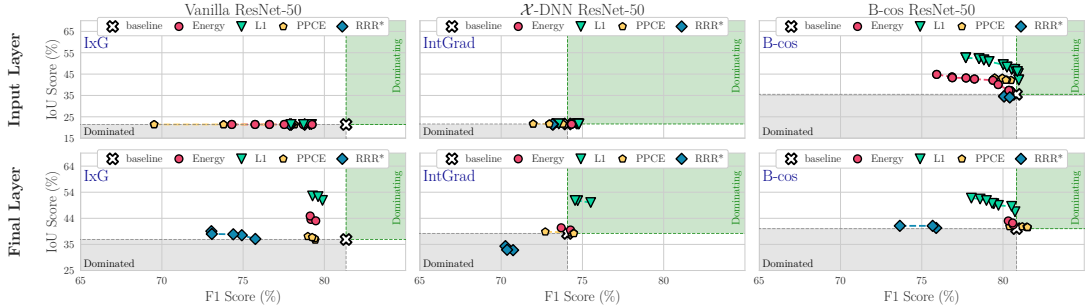
D.4.2 Optimising B-cos Attributions

Training for optimising the localisation of attributions (Equation (7.1)) requires backpropagating through the attribution maps, which implies that they need to be differentiable. While B-cos attributions [BFS22b] as formulated are mathematically differentiable, the original implementation[3] [BSFS24] for computing them involves detaching the dynamic weights from the computational graph, which prevents them from being used for optimisation. In this work, to use them for model guidance, we develop a twice-differentiable implementation of B-cos attributions.

D.5 Full Results

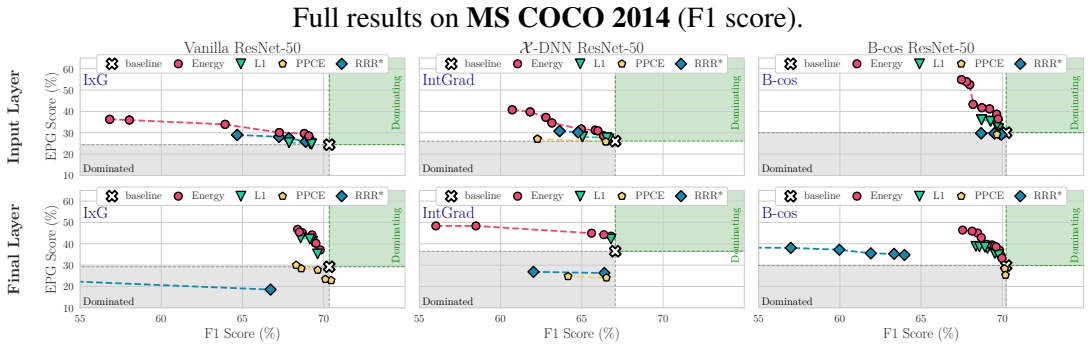


(a) **EPG vs. F1.**

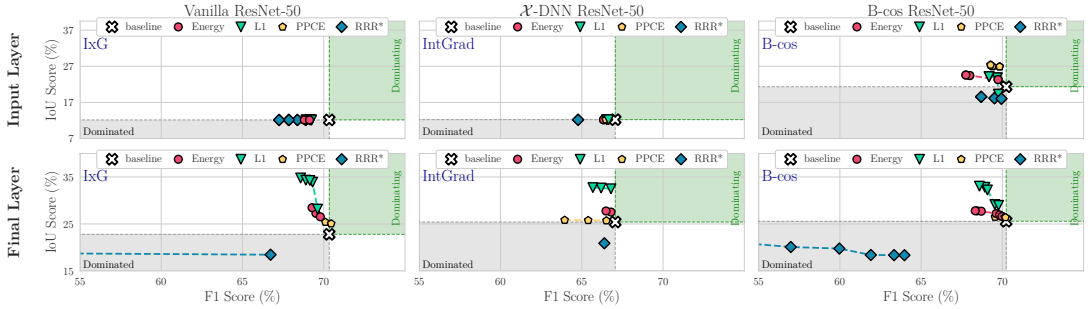


(b) **IoU vs. F1.**

Figure D.15: **EPG (a) and IoU (b) vs. F1 on VOC,** for different losses (**markers**) and models (**columns**), optimised at different layers (**rows**); additionally, we show the performance of the baseline model before fine-tuning and demarcate regions that strictly dominate (are strictly dominated by) the baseline performance in green (grey). For each configuration, we show the Pareto fronts (cf. Figure 7.4) across regularisation strengths λ_{loc} and epochs (cf. Section 7.3 and Figure 7.4). We find the Energy loss to give the best trade-off between EPG and F1, whereas the L_1 loss (especially at the final layer) provides the best trade-off between IoU and F1. We further find these results to be consistent across datasets, see Figure D.16.

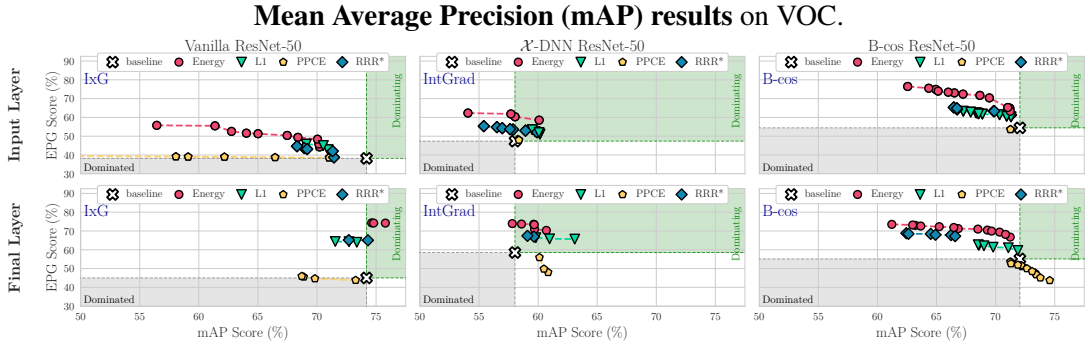


(a) **EPG vs. F1.**

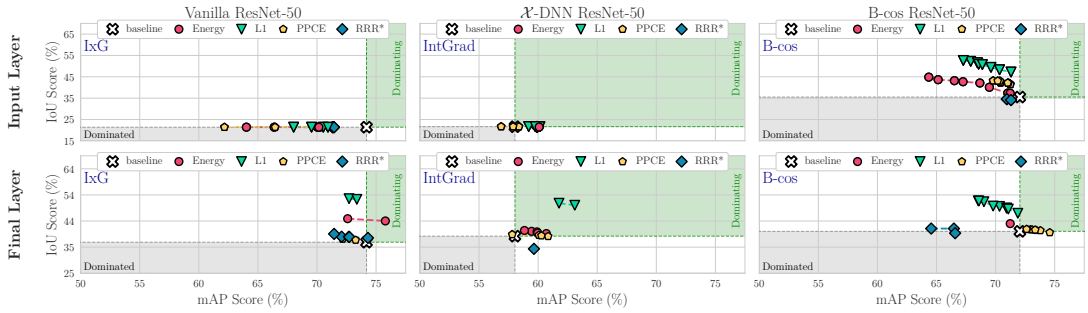


(b) **IoU vs. F1.**

Figure D.16: **EPG (a) and IoU (b) vs. F1 on COCO,** for different losses (**markers**) and models (**columns**), optimised at different layers (**rows**); additionally, we show the performance of the baseline model before fine-tuning and demarcate regions that strictly dominate (are strictly dominated by) the baseline performance in green (grey). For each configuration, we show the Pareto fronts (cf. Figure 7.4) across regularisation strengths λ_{loc} and epochs (cf. Section 7.3 and Figure 7.4). We find the Energy loss to give the best trade-off between EPG and F1, whereas the L_1 loss (especially at the final layer) provides the best trade-off between IoU and F1. We further find these results to be consistent across datasets, see Figure D.15.

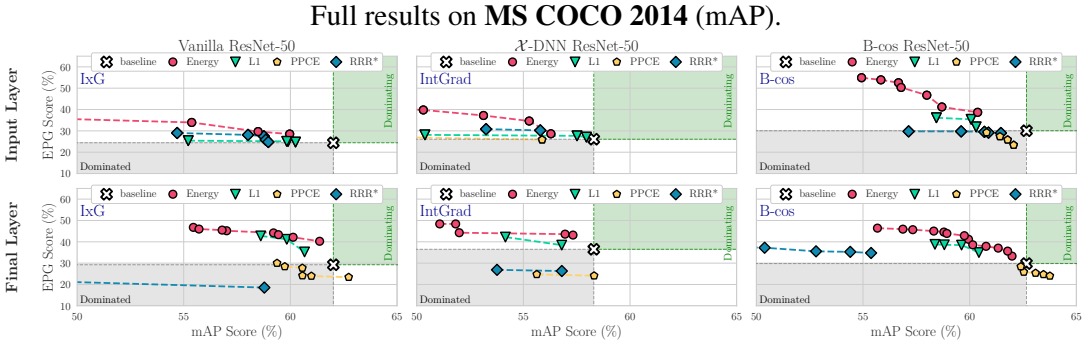


(a) **EPG vs. mAP.**



(b) **IoU vs. mAP.**

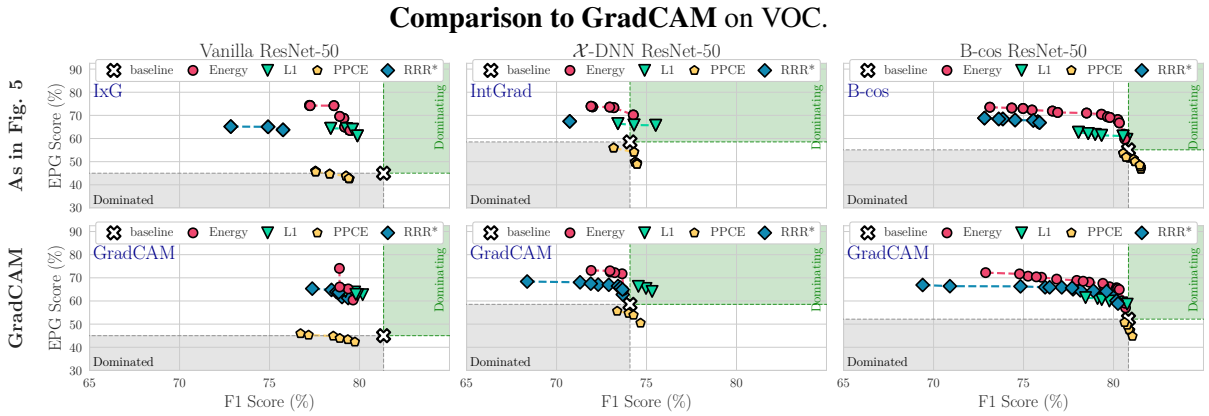
Figure D.17: **Quantitative comparison of EPG and IoU vs. mAP scores for VOC.** To ensure that the trends observed and described in Chapter 7 generalise beyond the F1 metric, in this figure we show the EPG and IoU scores plotted against the mAP metric. In general, we find the results obtained for the mAP metric to be highly consistent with the previously shown results for the F1 metric, see Figure D.15. E.g., across all configurations, we find the Energy to yield the highest gains in EPG score, whereas the L_1 loss provides the best trade-offs with respect to the IoU metric. These results are further also consistent with those observed on COCO, see Figure D.18.



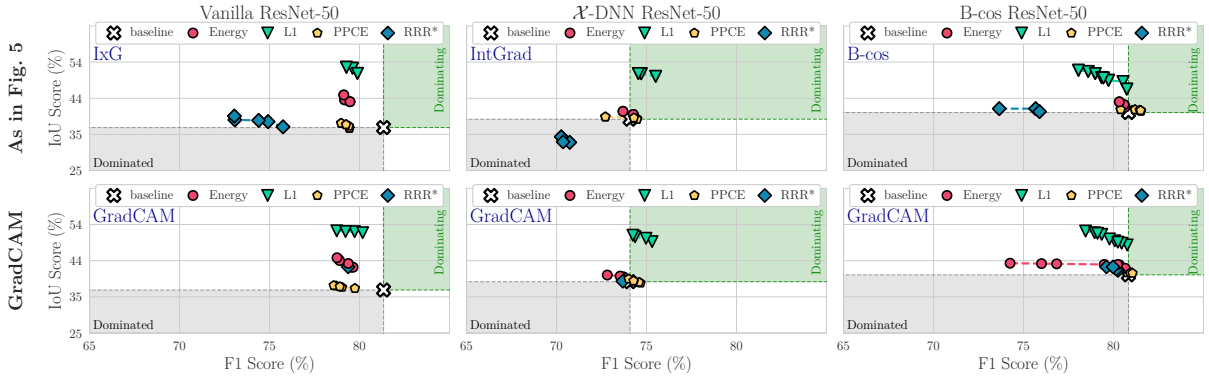
(a) **EPG vs. mAP.**

(b) **IoU vs. mAP.**

Figure D.18: **Quantitative comparison of EPG and IoU vs. mAP scores for COCO.** To ensure that the trends observed and described in Chapter 7 generalise beyond the F1 metric, in this figure we show the EPG and IoU scores plotted against the mAP metric. In general, we find the results obtained for the mAP metric to be highly consistent with the previously shown results for the F1 metric, see Figure D.16. E.g., across all configurations, we find the Energy to yield the highest gains in EPG score, whereas the L_1 loss provides the best trade-offs with respect to the IoU metric. These results are further also consistent with those observed on VOC, see Figure D.17.

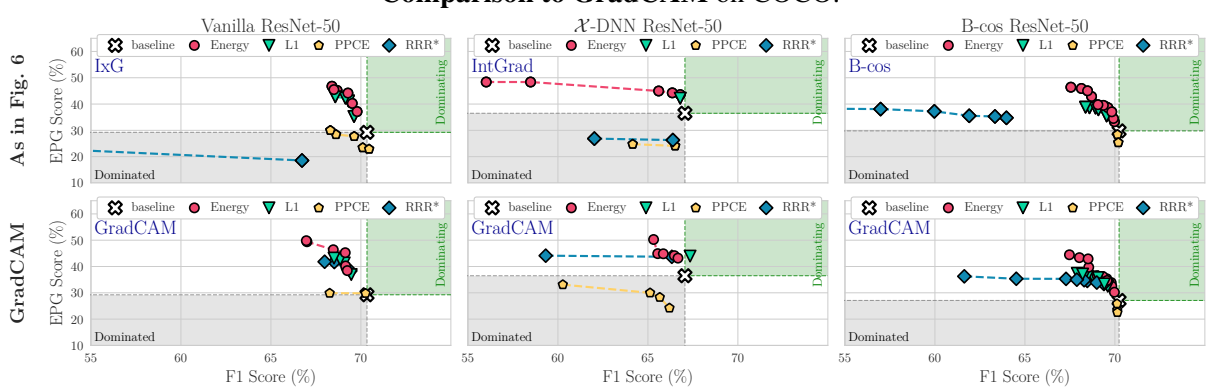


(a) **EPG vs. F1.**

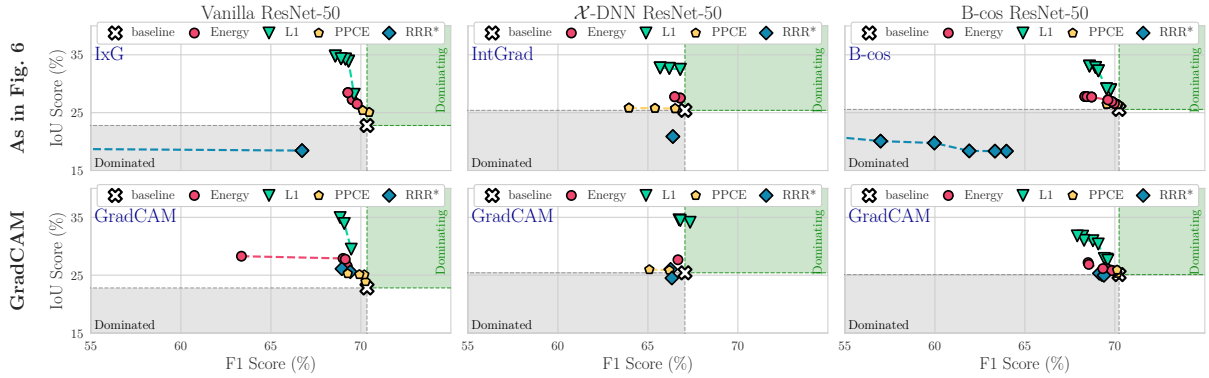


(b) **IoU vs. F1.**

Figure D.19: **Quantitative results using GradCAM on VOC.** We show EPG **(a)** and IoU **(b)** scores vs. F1 scores for all localisation losses and models using GradCAM at the final layer (**bottom rows** in (a)+(b) and compare it to the results shown in Chapter 7 (**top rows**). As expected, GradCAM performs very similarly to IxG (Vanilla) and IntGrad ($\mathcal{X}$-DNN) used at the final layer—in particular, note that for ResNet-50 architectures, IxG and IntGrad are very similar to GradCAM for Vanilla and $\mathcal{X}$-DNN models respectively (see Section D.2.2). Similarly, we find GradCAM to also perform comparably to the B-cos explanations when used at the final layer; for results on COCO, see Figure D.20.



(a) **EPG vs. F1.**



(b) **IoU vs. F1.**

Figure D.20: **Quantitative results using GradCAM on COCO.** We show EPG **(a)** and IoU **(b)** scores vs. F1 scores for all localisation losses and models using GradCAM at the final layer (**bottom rows** in (a)+(b) and compare it to the results shown in Chapter 7 (**top rows**). As expected, GradCAM performs very similarly to IxG (Vanilla) and IntGrad ($\mathcal{X}$-DNN) used at the final layer—in particular, note that for ResNet-50 architectures, IxG and IntGrad are very similar to GradCAM for Vanilla and $\mathcal{X}$-DNN models respectively (see Section D.2.2). Similarly, we find GradCAM to also perform comparably to the B-cos explanations when used at the final layer; for results on VOC, see Figure D.19.

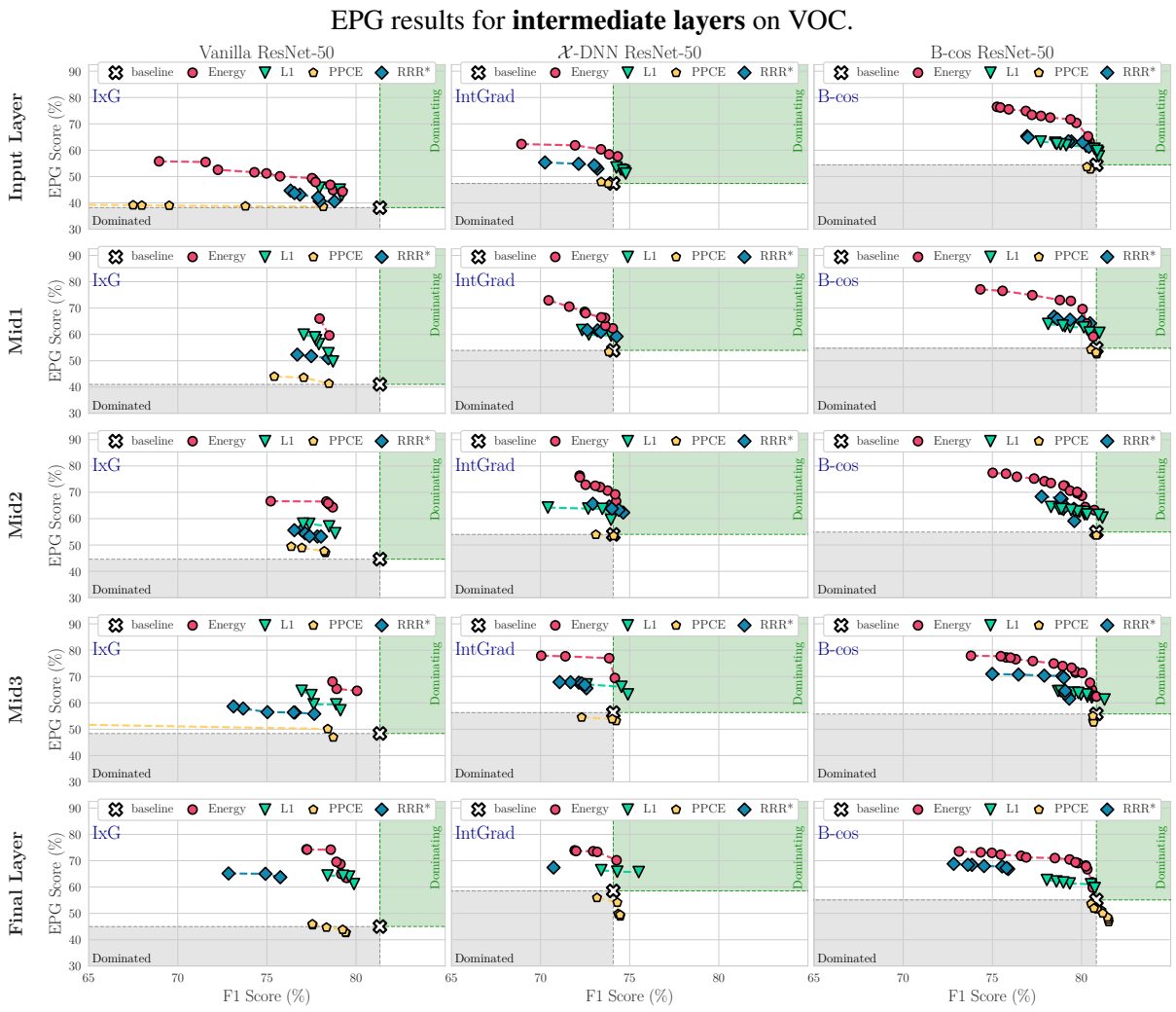
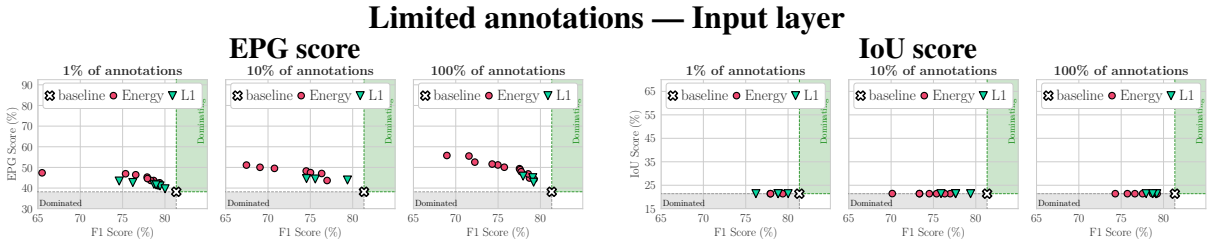
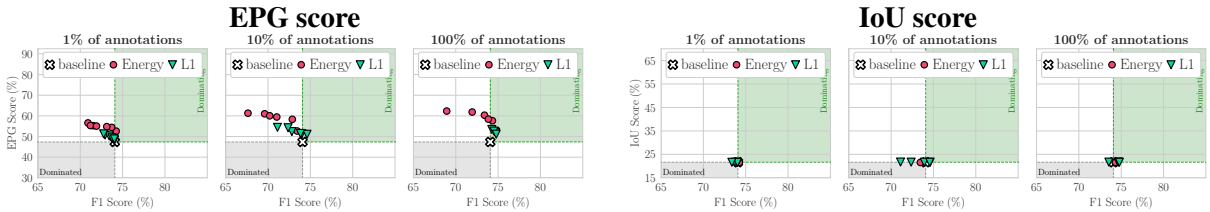


Figure D.21: **Intermediate layer results comparing EPG vs. F1.** We compare the effectiveness of model guidance at varying network depths (**rows**) for each attribution method and model (**columns**) across localisation loss functions. For the B-cos model, we find similar trends at all network depths, with the Energy localisation loss outperforming all other losses. For the Vanilla and $\mathcal{X}$-DNN models, the Energy loss similarly performs the best, but we also observe improved performance across losses when optimising at deeper layers of the network. Results for IoU can be found in Figure D.22.

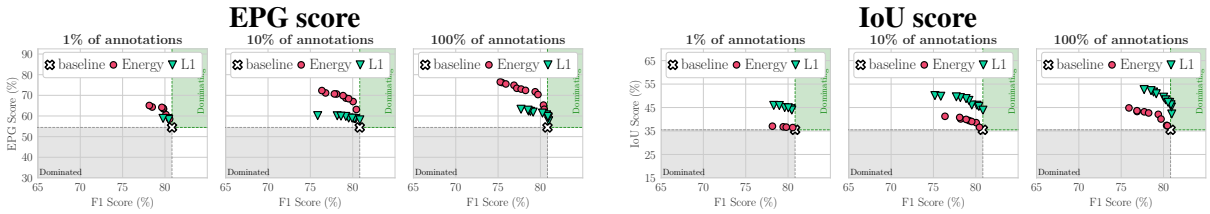
Figure D.22: **Intermediate layer results comparing IoU vs. F1.** We compare the effectiveness of model guidance at varying network depths (**rows**) for each attribution method and model (**columns**) across localisation loss functions. We find similar trends across all configurations, with the L_1 loss outperforming all other losses. For the Vanilla and $\mathcal{X}$-DNN models, we observe improved performance across losses when optimising at deeper layers of the network, whereas the results seem very stable for the B-cos models. For EPG results, see Figure D.21.



(a) **Vanilla ResNet-50**

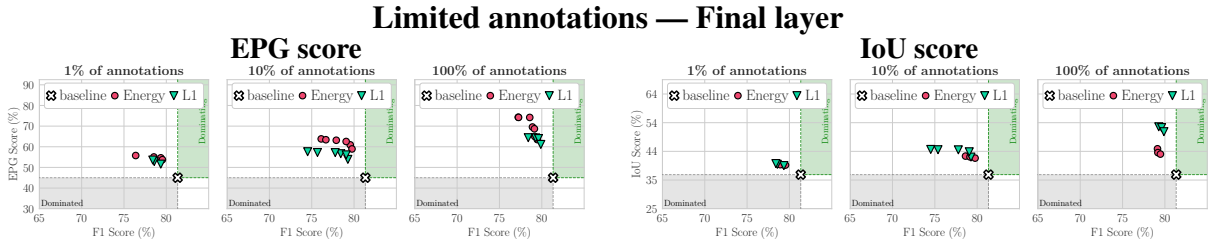


(b) $\mathcal{X}$**-DNN ResNet-50**

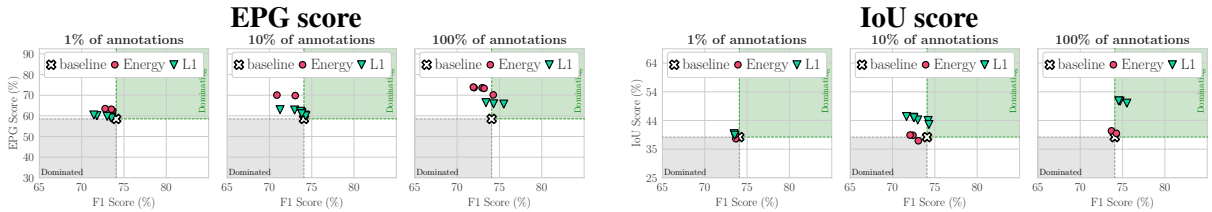


(c) **B-cos ResNet-50**

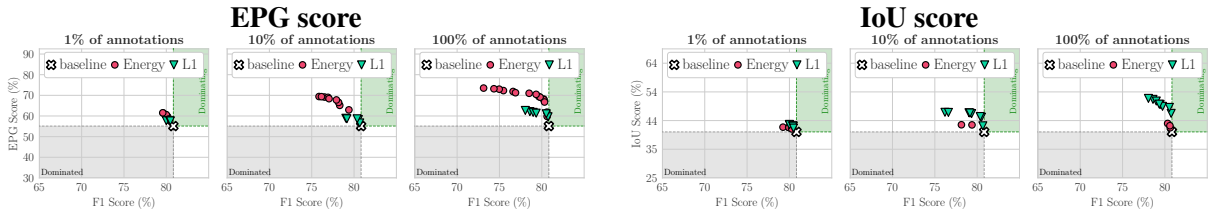
Figure D.23: **EPG and IoU scores for model guidance at the input layer with a limited number of annotations.** We show EPG vs. F1 (**left**) and IoU vs. F1 (**right**) for all models, optimised with the Energy and L_1 localisation losses, when using $\{1\%, 10\%, 100\%\}$ training annotations. We find that model guidance is generally effective even when training with annotations for a limited number of images. While the performance slightly worsens when using 1% annotations, using just 10% annotated images yields similar gains to using a fully annotated training set. Results at the final layer can be found in Figure D.24.



(a) **Vanilla ResNet-50**

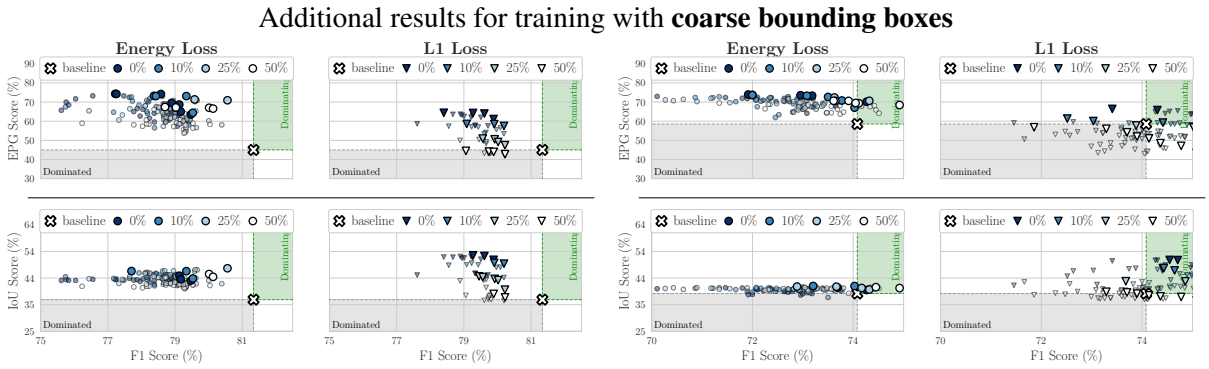


(b) $\mathcal{X}$**-DNN ResNet-50**

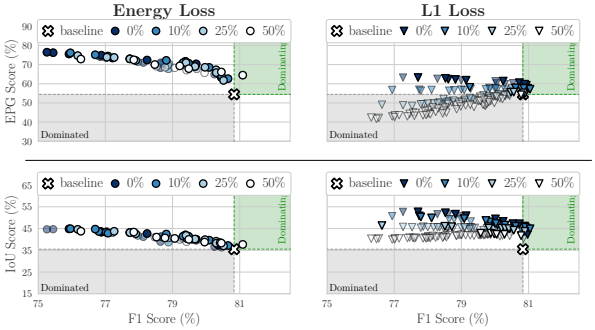


(c) **B-cos ResNet-50**

Figure D.24: **EPG and IoU scores for model guidance at the final layer with a limited number of annotations.** We show EPG vs. F1 (**left**) and IoU vs. F1 (**right**) for all models, optimised with the Energy and L_1 localisation losses, when using $\{1\%, 10\%, 100\%\}$ training annotations. We find that model guidance is generally effective even when training with annotations for a limited number of images. While the performance worsens when using 1% annotations, using just 10% annotated images yields similar gains to using a fully annotated training set. Results at the input layer can be found in Figure D.23.



(a) **Vanilla ResNet-50 @ Final.** (b) $\mathcal{X}$**-DNN ResNet-50 @ Final.**



(c) **B-cos ResNet-50 @ Input.**

Figure D.25: **Coarse bounding box results**. We show the impact of dilating bounding boxes during training for the **(a)** Vanilla and **(b)** $\mathcal{X}$-DNN, and **(c)** B-cos models. Similar to the results seen with B-cos models (c), we find that the Energy localisation loss is generally robust to coarse annotations, while the effectiveness of guidance with the L_1 localisation loss worsens as the extent of coarseness increases.

APPENDIX — EXPLANATION-ENHANCED KNOWLEDGE DISTILLATION

E

In this supplement to our work on explanation-enhanced knowledge distillation (e^2KD), we provide:

E.1 Additional Qualitative Results

E.1.1 Learning the 'right' Features

We provide qualitative results on the Waterbirds-100 dataset [SKHL20, PDN+22] and show GradCAM explanations [SCD+17] for standard models and B-cos explanations for B-cos models [BFS22b, BSFS24]. In Figure E.1, we show explanations for in-distribution (i.e. 'Landbird on Land' and 'Waterbird on Water') test samples, and in Figure E.2 we show them for out-of-distribution samples (i.e. 'Landbird on Water' and 'Waterbird on Land'). Corresponding quantitative results can be found in Section E.2.2.

In Figure E.1, we see that the explanations of the teacher and vanilla KD student may significantly differ (for both standard and B-cos models): while the teacher focuses on the bird, the student uses spuriously correlated input-features (i.e. background). We observe that e^{2}KD is successfully promoting explanation similarity and keeping the student 'right for right reasons'. While in Figure E.1 (i.e. in-distribution data) all models correctly classify the samples despite the difference in their focus, in Figure E.2 (i.e. out-of-distribution) we observe that the student trained with e^{2}KD is able to arrive at the correct prediction, whereas the vanilla KD student wrongly classifies the samples based on the background type.

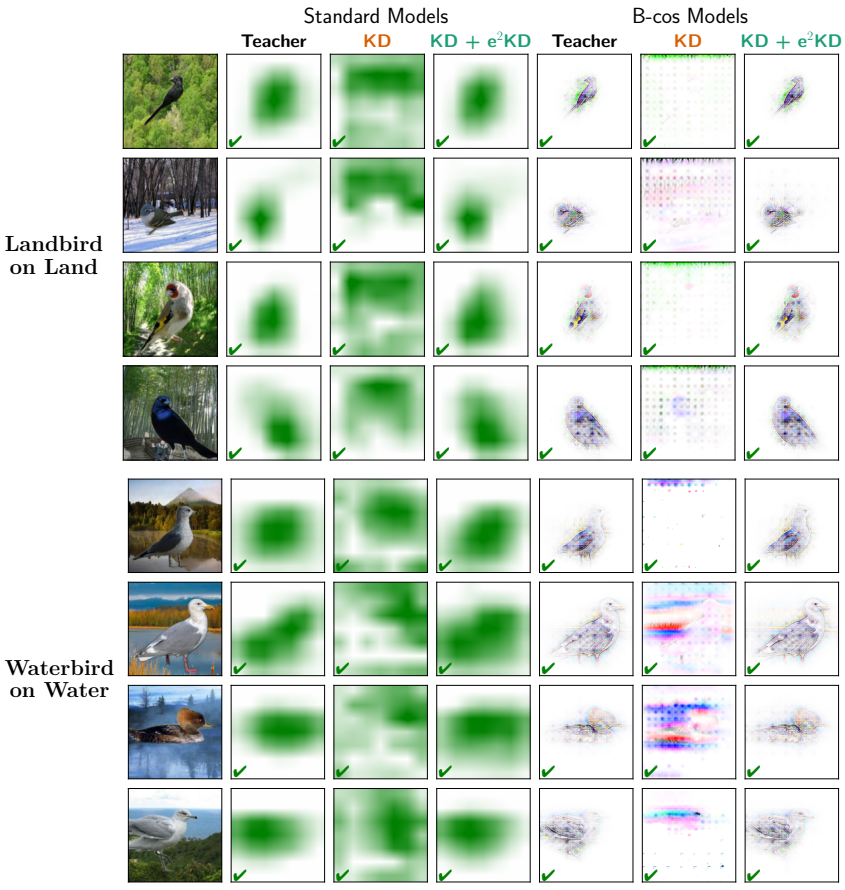


Figure E.1: **In-distribution samples for distillation on biased data using the Waterbirds-100 dataset**. We show explanations for both standard models (cols. 2-4) and B-cos models (cols. 5-7), given both in-distribution groups: 'Landbird on Land' (**top half**) and 'Waterbird on Water' (**bottom half**). We find that e^{2}KD approach (col. 4 and 7) is effective in preserving the teacher's focus (col. 2 and 5) to the bird instead of the background as opposed to vanilla KD (col. 3 and 6). Correct and incorrect predictions marked by ✓ and ✗ respectively.

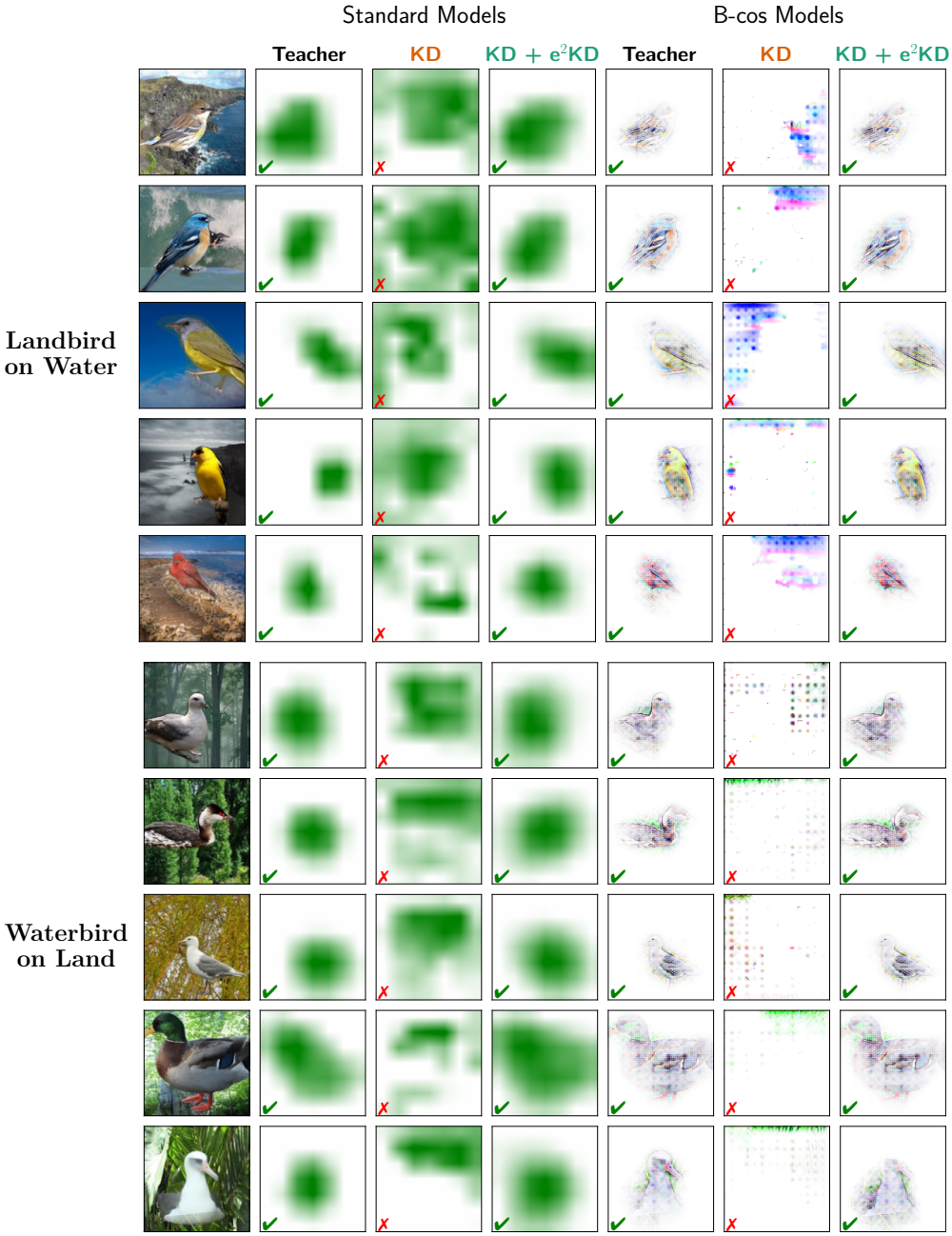


Figure E.2: **Out-of-distribution samples for distillation on biased data using the Waterbirds-100 dataset**. We show explanations for standard (cols. 2-4) and B-cos models (cols. 5-7), for the out-of-distribution groups 'Landbird on Water' (**top half**) and 'Waterbird on Land' (**bottom half**). e^2KD (col. 4 and 7) is effective in preserving the teacher's focus (col. 2 and 5), leading to higher robustness to distribution shifts than when training students via vanilla KD. Correct and incorrect predictions marked by ✓ and ✗ respectively.

E.1.2 Maintaining Focused Explanations

In this section we provide additional qualitative examples for experiments on PASCAL VOC 2007 (see Section 8.2.3.1). We provide samples for all of the 20 classes in the PASCAL VOC 2007 dataset (every row in Figures E.3 and E.4). Across all classes we ovserve that the student trained with e^2KD maintains focused explanations on the class-specific input-features, whereas the student trained with vanilla KD may often focus on the background.

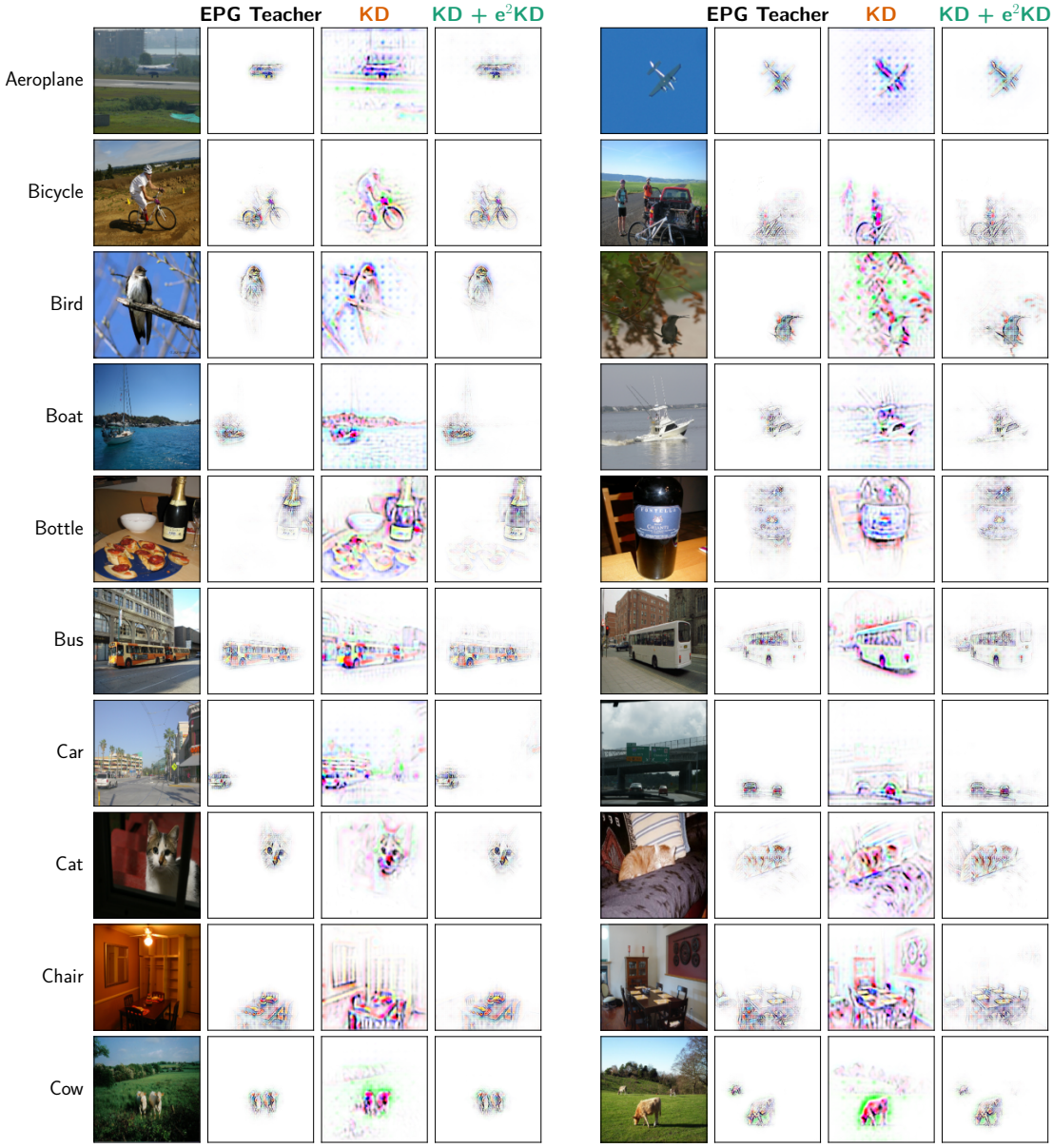


Figure E.3: **Maintaining focused explanations (classes 1-10)**: Similar to Figure 8.3 in Chapter 8, here we show qualitative difference of explanations. Each row shows two samples per class (for classes 11-20 see Figure E.4). We find that explanations of the student trained with e^2KD (subcol. 4 on both sides) is significantly closer to the teacher's (subcol. 2), whereas vanilla KD students also focus on the background (subcol. 3). Samples were drawn from the test set with all models having correct predictions.

Figure E.4: **Maintaining focused explanations (classes 11-20)**: Similar to Figure 8.3 in Chapter 8, here we show qualitative difference of explanations. Each row shows two samples per class (for classes 1-10 see Figure E.3). We find that explanations of the student trained with e^2KD (subcol. 4 on both sides) is significantly closer to the teacher's (subcol. 2), whereas vanilla KD students also focus on the background (subcol. 3). Samples were drawn from test set with all models having correct predictions.

E.1.3 Distilling Architectural Priors

In this section, we provide additional qualitative samples for Section 8.2.3.2 in Chapter 8, where we distill a B-cos CNN to a B-cos ViT. Looking at Figure E.5, one can immediately observe the difference in interpretability of the B-cos ViT explanations when trained with e^{2}KD vs. vanilla KD. Additionally, following what was reported in Figure 8.5, one can see that the ViT trained with e^{2}KD, similar to its CNN Teacher, maintains consistent explanations under shift, despite its inherent tokenisation, whereas the explanations from vanilla KD significantly differ (compare every odd row to the one below).

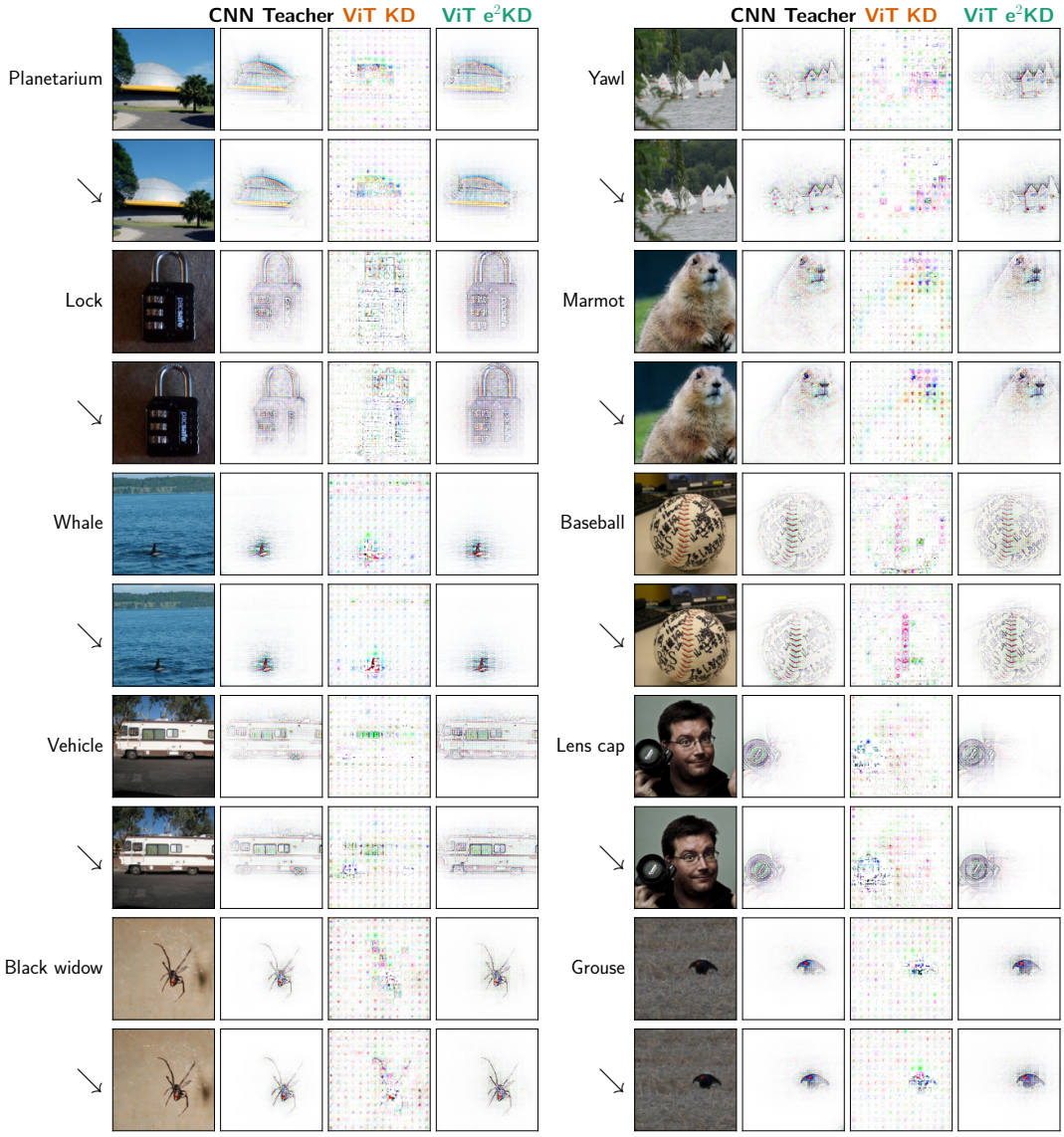


Figure E.5: **Qualitative results on distilling B-cos DenseNet-169 to B-cos ViT$_{\text{Tiny}}$.** We see that explanations of the e^{2}KD ViT student (col. 4 on both sides) is significantly more interpretable than vanilla KD student (col. 3 on both sides), and very close to the teacher's explanations (col. 2 on both sides). We also shift every image diagonaly to the bottom right by 8 pixels and show the explanations for the same class (rows indicated by ↘). We see that the explanations of the ViT student trained with e^{2}KD are equivariant under such a shift. In contrast, the ViT student from vanilla KD is sensitive to such shifts and the explanations change significantly.

E.2 Additional Quantitative Results

E.2.1 Reproducing Previously Reported Results for Prior Work.

Since we use prior work on new settings, namely ImageNet with limited data, Waterbirds-100, and B-cos models, we reproduce the performance reported in the original works in the following. In particular, in Table E.1, we report the results obtained by the training 'recipes' followed by prior work to validate our implementation and enable better comparability with previously reported results.

For this, we distilled the standard ResNet-34 teacher to a ResNet-18 on ImageNet for 100 epochs, with an initial learning rate of 0.1, decayed by 10% every 30 epochs. We used SGD with momentum of 0.9 and a weight-decay factor of 1e-4. For AT and ReviewKD, we followed the respective original works and employed weighting coefficients of $\lambda \in \{1000.0, 1.0\}$ respectively. For CAT-KD, we used $\lambda \in \{1.0, 5.0, 10.0\}$ after identifying this as a reasonable range in preliminary experiments.

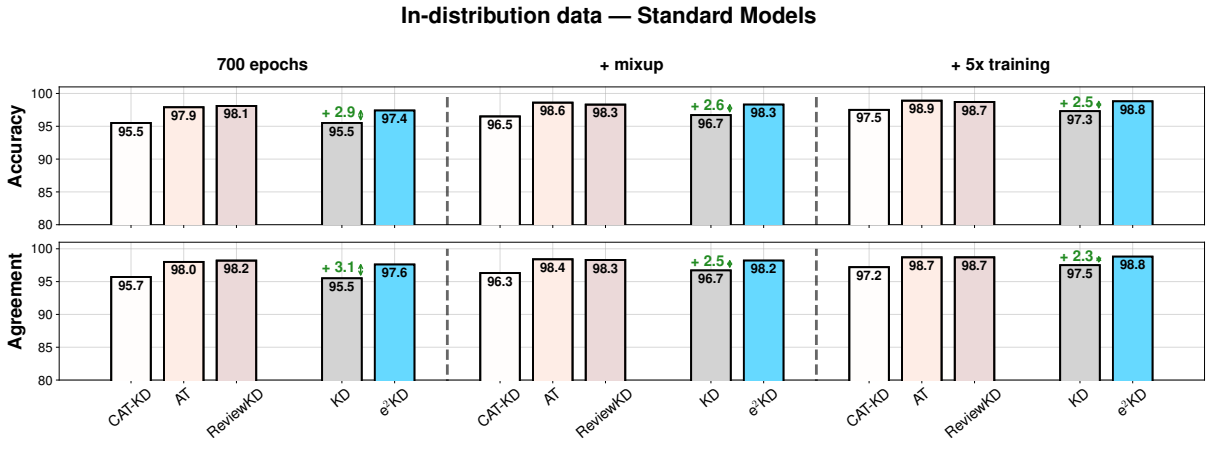
Following [HVD15], we also report the results for KD in which the cross-entropy loss with respect to the ground truth labels is used in the loss function. We were able to reproduce the reported numbers by a close margin. Our numbers are also comparable to the torchdistill's reproduced numbers [Mat21], see Table E.1. We see that e^{2}KD again improves both accuracy and agreement of vanilla KD (agreement 80.2→80.5). Also note that the vanilla KD baseline significantly improves once we used the longer training recipe from [BZR+22] in Table 8.1 in Chapter 8 (accuracy 70.6→71.8; agreement 80.2→81.2).

Standard Models Teacher ResNet-34 Accuracy 73.3%	Accuracy	Agreement	Reported Accuracy	torchdistill Accuracy
KD [HVD15] (with cross-entropy)	71.0	79.7	70.7	71.4
AT [ZK17]	70.2	78.3	70.7	70.9
ReviewKD [CLZJ21]	71.6	80.1	71.6	71.6
CAT-KD [GYLL23]	71.0	80.1	71.3	-
KD	70.6	80.2	-	-
+ e^{2}KD (GradCAM)	**70.7** (+ 0.1)	**80.5** (+ 0.3)	-	-

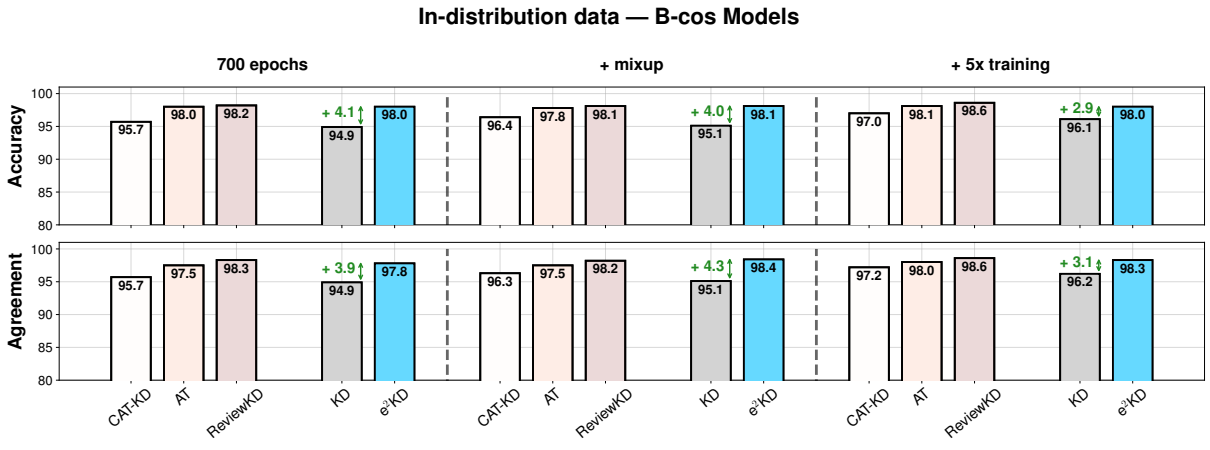
Table E.1: **Distilling Standard ResNet-34 to ResNet-18 for reproducing prior work.** We verify our implementation of prior work by distilling them in the 100 epoch setting used in [ZK17, CLZJ21, GYLL23]. We observe that our accuracy is very close to the reported one and the reproduced numbers by torchdistill [Mat21]. We also see that e^{2}KD, similar to Table 8.1, improves accuracy and agreement of vanilla KD.

E.2.2 Full Results on Waterbirds — B-cos and Conventional Models on In- and Out-of-distribution Data

In this section we provide complete quantitative results on Waterbirds-100 dataset [SKHL20, PDN+22], for both standard and B-cos models. In Figure E.6, we report in-distribution accuracy and agreement. We observe that all models are performing well on in-distribution data (lowest test-accuracy is 94.9% for standard and 95.5% for B-cos models.). Nevertheless, e^2KD is again consistently providing gains over vanilla KD. More importantly however, for the out-of-distribution samples, as reported in Figure E.7, we observe that e^2KD offers even larger accuracy and agreement gains over vanilla KD for both standard and B-cos models. Corresponding qualitative results, for the 700 epoch experiments, can be found in Figure E.1, for in-distribution, and Figure E.2 for out-of-distribution samples.

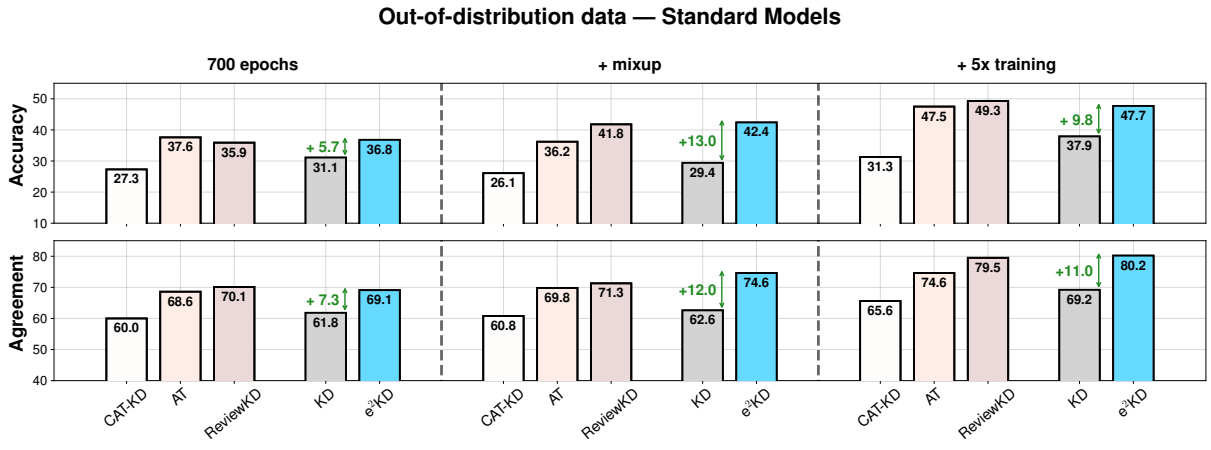


(a) In-distribution — Standard models — Teacher Acc. : 99.0%

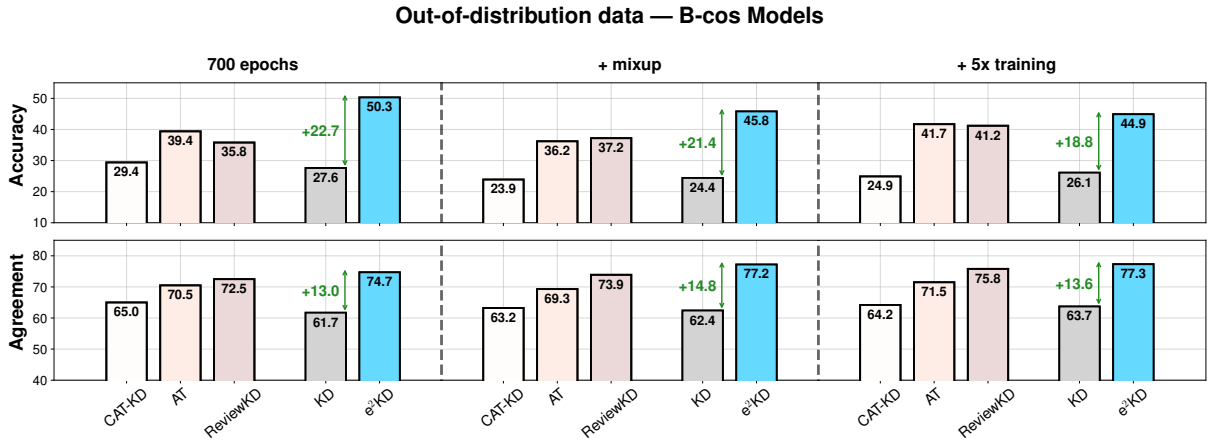


(b) In-distribution — B-cos models — Teacher Acc. : 98.8%

Figure E.6: **KD on *in-distribution data* for standard (a) and B-cos (b) models on Waterbirds-100.** We show accuracy (**top rows**) and agreement (**bottom rows**) when distilling from ResNet-50 teachers to ResNet-18 students with various KD approaches. As for out-of-distribution data (see Figure E.7), we find significant and consistent gains in accuracy and agreement on in-distribution for both model types.



(a) Out-of-distribution — Standard models — Teacher Acc. : 61.2%



(b) Out-of-distribution — B-cos models — Teacher Acc. : 55.2%

Figure E.7: **KD on *out-of-distribution data* for standard (a) and B-cos (b) models on Waterbirds-100;** note that (a) is the same as Figure 8.4 in Chapter 8 and repeated here for easier reference. We show accuracy (**top rows**) and agreement (**bottom rows**) when distilling from ResNet-50 teachers to ResNet-18 students with various KD approaches. Similar as for the standard models (a), we find that e^2KD also significantly increases both accuracy and agreement for B-cos models. For results on in-distribution data, please see Figure E.6.

E.3 Implementation Details

In this section, we provide additional implementation details. In Section E.3.1, we provide a detailed description of our training setup, including hyperparameters used in each setting. In Section E.3.2, we describe how we adapt prior approaches that were proposed for conventional deep neural networks to B-cos models. Code for all the experiments will be made available.

E.3.1 Training Details

In this section, we first provide the general setup which, unless specified otherwise, is shared across all of our experiments. Afterwards, we describe dataset-specific details in Sections E.3.1.1 to E.3.1.3, for each dataset and experiment.

Standard Networks. As mentioned in Section 8.2 in Chapter 8, we follow the recipe from [BZR+22]. For standard models, we use the AdamW optimiser [KB15] with a weight-decay factor of 10^{-4} and a cosine-annealing learning-rate scheduler [LH17] with an initial learning-rate of 0.01, reached with an initial warmup for 5 epochs. We clip gradients by norm at 1.0.

B-cos Networks. We use the latest implementations for B-cos models [BSFS24]. Following [BFS22b, BSFS24], we use the Adam optimiser [KB15] and do not apply weight-decay. We use a cosine-annealing learning-rate scheduler [LH17] with an initial learning-rate of 10^{-3}, reached with an initial warmup for 5 epochs. Following [BSFS24], we clip gradients using adaptive gradient clipping (AGC) [BDSS21b].

Unless specified otherwise, across all models and datasets we use random crops and random horizontal flips as data augmentation during training, and at test time we resize the images to 256 (along the smaller dimension) and apply centre crop of (224, 224). We use PyTorch [PGM+19] and PyTorch Lightning [FT19] for all of our implementations.

E.3.1.1 *ImageNet Experiments*

For experiments on the full ImageNet dataset [DDS+09], we use a batch size of 256 and train for 200 epochs. For limited-data experiments we keep the number of steps same across both settings (roughly 40% total steps compared to full-data): when using 50 shots per class, we set the batch size to 32 and train for 250 epochs, and when having 200 shots, we use a batch size of 64 and train for 125 epochs. We use the same randomly selected shots for all limited-data experiments. For Data-free experiments (Table 8.5 in Chapter 8), following [BZR+22], we used equal number of training steps for both SUN→IMN (125 epochs with a batch size of 128) and IMN→SUN (21 epochs with a batch size of 256).

For the pre-trained teachers, we use the Torchvision checkpoints[1] [mc16] for standard models and available checkpoints for B-cos models[2] [BSFS24]. For all ImageNet experiments, we pick the best checkpoint and loss coefficients based on a held-out subset of the standard train set, which has 50 random samples per class. The results are then reported on the entire official validation set. We use the following parameters for each method:

[1] https://pytorch.org/vision/stable/models.html
[2] https://github.com/B-cos/B-cos-v2

Standard Networks (Table 8.1)

KD	$\tau \in [1, 5]$
e^2KD	$\tau \in [1, 5]$, $\lambda \in [1, 5, 10]$
AT	$\lambda \in [10, 100, 1000, 10000]$
ReviewKD	$\lambda \in [1, 5]$
CAT-KD	$\lambda \in [1, 5, 10]$

B-cos ResNet-34 Teacher (Table 8.2)

KD	$\tau \in [1, 5]$
e^2KD	$\tau \in [1, 5]$, $\lambda \in [1, 5]$
AT	$\lambda \in [10, 100, 1000]$
ReviewKD	$\lambda \in [1, 5]$
CAT-KD	$\lambda \in [1, 5]$

B-cos DenseNet-169 Teacher (Tables 8.3 and 8.5)

KD	$\tau \in [1, 5]$
e^2KD	$\tau \in [1, 5]$, $\lambda \in [1, 5, 10]$
❄ KD	$\tau \in [1, 5]$
❄ e^2KD	$\tau \in [1, 5]$, $\lambda \in [0.2, 1, 5, 10]$
Data-free KD	$\tau \in [1, 5]$
Data-free e^2KD	$\tau \in [1, 5]$, $\lambda \in [1, 5, 10]$

B-cos DenseNet-169 to ViT Student (Figure 8.5)

KD	$\tau \in [1, 5]$
e^2KD	$\tau \in [1]$, $\lambda \in [1, 5, 10]$

For ViT students we trained for 150 epochs, and following [BSFS24], we used 10k warmup steps, and additionally used RandAugment [CZSL20b] with magnitude of 10.

E.3.1.2 *Waterbirds Experiments*

For the Waterbirds [SKHL20] experiments, we use the 100% correlated data generated by [PDN+22] (i.e. Waterbirds-100). We use the provided train, validation and test splits. Since the data is imbalanced (number of samples per class significantly differ), within each sweep we pick the last-epoch checkpoint with best *overall* validation accuracy (including both in-distribution and out-of-distribution samples). We use batch size of 64. For experiments with MixUp, we use α=1. The pre-trained guided teachers were obtained from [RBPAS23]. For applying AT and ReviewKD between the ResNet-50 teacher and ResNet-18 student, we used the same configuration from a ResNet-34 teacher, since they have the same number of blocks.

We tested the following parameters for each method:

Standard models (Figure 8.4 and Section E.2.2)

KD	$\tau \in [1, 5]$
e^2KD	$\tau \in [1, 5]$, $\lambda \in [1, 5, 10, 15]$
AT	$\lambda \in [10, 100, 1000]$
ReviewKD	$\lambda \in [1, 5, 10, 15]$
CAT-KD	$\lambda \in [1, 5, 10, 15]$

B-cos models (Figure 8.2 and Section E.2.2)

KD	$\tau \in [1, 5]$
e^2KD	$\tau \in [1]$, $\lambda \in [1, 5, 10]$
AT	$\lambda \in [10, 100, 1000]$
ReviewKD	$\lambda \in [1, 5, 10]$
CAT-KD	$\lambda \in [1, 5, 10]$

E.3.1.3 *Pascal VOC Experiments*

We use the 2012 release of PASCAL VOC dataset [EVGW+12]. We randomly select 10% of the train samples as validation set and report results on the official test set. We use batch size of 64 and train for 150 epochs. The pre-trained guided teachers were obtained from [RBPAS23]. Since we are using VOC as a *multi-label* classification setting, we replace the logit loss from Equation (8.1) in Chapter 8 with the logit loss recently introduced by [YXZ+23]:

$$\mathcal{L}_{MLD} = \tau \sum_{j=1}^{c} D_{\mathrm{KL}} \left(\left[\psi_j \left(\frac{z_T}{\tau} \right), 1 - \psi_j \left(\frac{z_T}{\tau} \right) \right] || \left[\psi_j \left(\frac{z_S}{\tau} \right), 1 - \psi_j \left(\frac{z_S}{\tau} \right) \right] \right) . \tag{E.1}$$

Here, ψ is the sigmoid function and $[.,.]$ concatenates values into a vector. Note that the original loss from [YXZ+23] does not have a temperature parameter τ (i.e. $\tau = 1$). For consistency with other experiments, here we also included a temperature factor. When reporting the final results on the test set, we resize images to (224, 224) and do not apply centre crop. For the EPG and IoU metrics, we use the

implementation from [RBPAS23]. For the IoU metric, we use threshold of 0.05. We tested the following parameters for each method:

B-cos models (Table 8.4)
KD $\tau \in [1, 5]$,
e^{2}KD $\tau \in [1, 5]$, $\lambda \in [1, 5, 10]$

E.3.2 Adapting Prior Feature-based Methods for B-cos Models

While prior feature-based KD methods have been mainly introduced for conventional networks, in Table 8.2 we additionally tested them on B-cos Networks. We applied them with the same configuration that they were originally introduced as for ResNet-34 [HZRS16] teacher and ResNet-18 student, with minor adjustments. Specifically, since B-cos Networks also operate on negative subspace, we did not apply ReLU on the intermediate tensors in AT. For ReviewKD, since the additional convlution and norm layers between the teacher and student are only needed to convert intermediate representations, we used standard convolution and BatchNorm and not B-cos specific layers. For AT, ReviewKD, and CAT-KD we replaced the cross-entropy loss, with the modified binary cross entropy from [BSFS24].

APPENDIX — TEMPERATURE SCHEDULES

F

In this supplement to our work on temperature schedules, we provide:

F.1 Pseudo-Code for Reproducibility of Cosine Schedule

Algorithm F.1: Cosine Schedule

```
import numpy as np
def get_temperature(epoch, T, τ_−=0.1, τ_+=1.0):
    # epoch: current epoch; T: period length
    return (τ_+ − τ_−) × (1 + np.cos(2 × np.pi × ϵ/T))/2 + τ_−
```

Insert Algorithm F.1 into your favourite contrastive learning framework to check it out!

F.2 Implementation Details

Evaluation Details. Following [JCMW21], we separate 5000 images for CIFAR10/100-LT as a validation set for each split. As we discussed in Chapter 9, the performance of the model depends on the relative position within a period T. Therefore we utilise the validation split to choose a checkpoint for further testing on the standard test splits for CIFAR10/100-LT. Precisely, for each dataset, we select the evaluation epoch for the checkpoint based only on the validation set of the first random split; the other splits of the same dataset are evaluated using the same number of epochs. Note that for ImageNet 100-LT there is no validation split and we select the last checkpoint as in [JCMW21]. For a fair comparison, we also reproduce the numbers from [JCMW21] in the same way.

Division Into Head, Mid, and Tail Classes. Following [JCMW21], we divide all the classes into three categories: head classes are with the most number of samples, tail classes are with the least number of samples and mid are the rest. In particular, for CIFAR10-LT for each split there are 4 head classes, 3 mid classes, and 3 tail classes; for CIFAR100-LT there are 34 head classes, 33 mid classes, 33 tail classes; for ImageNet 100-LT head classes are classes with more than 100 instances, tail classes have less than 20 instances per class, and mid are the rest.

F.3 Extended Results

Extension of Figure 9.3 In Figure F.1 we provide full results of kNN accuracy on CIFAR10 when the model is trained with different fixed τ values and with coarse binary supervision. Especially tail classes are improved by instance discrimination (small τ_{tail}).

Head-mid-tail Classes Evaluation. In the following, we present a detailed comparison of SimCLR and SimCLR+TS on head, mid, and tail classes on CIFAR10-LT in Table F.1, on CIFAR100-LT in Table F.2 and on ImageNet 100-LT in Table F.3. We observe consistent improvement for all evaluation metrics for all types of classes over the three datasets.

Influence of TS on Uniform vs Long-tailed Distributions. To further corroborate that TS particularly helpful for imbalanced data, we apply TS for the uniformly distributed data. In Table F.4, we can observe that the cosine schedule yields significant and consistent gains for the long-tailed version of CIFAR10 (CIFAR10-LT), but not for the uniform one (CIFAR10-Uniform). We assume that both head classes and tail classes for long-tail distribution should be expected to benefit from a better separation between the two: on the one hand, the tail classes form better clusters and are thus easier to classify based on their

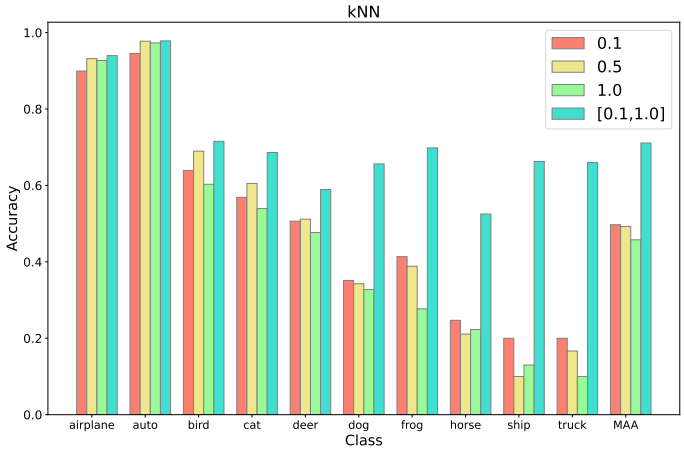


Figure F.1: **kNN accuracy for CIFAR10-LT trained with MoCo.** Comparison between $\tau{=}0.1$, $\tau{=}0.5$, $\tau{=}1.0$. [0.1, 1.0] denotes coarse supervision with $\tau_{\text{head}}{=}1.0$ and $\tau_{\text{tail}}{=}0.1$. MAA: mean average accuracy over all classes.

	CIFAR10-LT											
	kNN@1						kNN@10					
method	Head		Mid		Tail		Head		Mid		Tail	
SimCLR	84.93	± 3.44	54.08	± 4.24	32.14	± 7.44	88.03	± 3.32	53.76	± 4.80	29.52	± 9.44
SimCLR + TS	87.24	± 3.05	58.96	± 5.21	35.02	± 8.27	89.92	± 2.97	59.31	± 4.69	30.51	± 12.38
	FS LP						LT LP					
method	Head		Mid		Tail		Head		Mid		Tail	
SimCLR	76.38	± 5.24	63.20	± 2.95	62.60	± 3.63	89.52	± 3.15	56.98	± 4.74	29.88	± 8.11
SimCLR + TS	80.54	± 5.02	66.50	± 4.38	65.67	± 4.07	91.73	± 2.49	62.09	± 4.21	32.38	± 9.23

Table F.1: Detailed evaluation on CIFAR10-LT. Evaluation metrics: kNN@1,10, FS LP states for few-shot linear probe, and LT LP states for long-tail linear probe. We report the average performance with the standard deviation over three different random splits for different sets of classes: head, mid, and tail.

neighbours, on the other hand, the clusters of the head classes are 'purified', which should similarly improve performance. Weather, for the uniform distribution, we do not observe such influence of TS and the performance changes only marginally.

	CIFAR100-LT											
	kNN@1						kNN@10					
method	Head		Mid		Tail		Head		Mid		Tail	
SimCLR	53.87	± 2.12	24.56	± 1.51	7.26	± 0.39	58.46	± 1.79	22.15	± 1.47	2.83	± 0.61
SimCLR + TS	57.14	± 1.95	26.00	± 1.20	8.31	± 0.57	61.93	± 1.88	24.22	± 2.23	3.05	± 0.54
	FS LP						LT LP					
method	Head		Mid		Tail		Head		Mid		Tail	
SimCLR	33.48	± 1.24	24.25	± 2.12	19.12	± 1.35	62.19	± 1.80	26.56	± 1.46	3.92	± 0.46
SimCLR + TS	37.5	± 1.33	27.64	± 1.95	21.26	± 0.66	65.24	± 2.04	29.20	± 1.48	4.42	± 0.26

Table F.2: Detailed evaluation on CIFAR100-LT. Evaluation metrics: kNN@1,10, FS LP states for few-shot linear probe, and LT LP states for long-tail linear probe. We report the average performance with the standard deviation over three different random splits for different sets of classes: head, mid, and tail.

	ImageNet100-LT					
	kNN@1			kNN@10		
method	Head	Mid	Tail	Head	Mid	Tail
SimCLR	55.13	30.00	10.71	58.51	29.70	8.71
SimCLR + TS	57.23	30.26	13.14	60.41	29.53	10.14
	FS LP			LT LP		
method	Head	Mid	Tail	Head	Mid	Tail
SimCLR	51.79	36.77	30.29	67.59	36.47	9.43
SimCLR + TS	60.41	40.38	33.57	70.67	38.85	10.29

Table F.3: Detailed evaluation on ImageNet 100-LT. Evaluation metrics: kNN@1,10, FS LP states for few-shot linear probe, and LT LP states for long-tail linear probe. We report the average performance for different sets of classes: head, mid, and tail.

	CIFAR10-Uniform				CIFAR10-LT			
method	kNN@1	kNN@10	FS LP	LT LP	kNN@1	kNN@10	FS LP	LT LP
MoCo	83.47	84.87	**90.19**	**87.70**	63.00	64.10	68.89	63.99
MoCo + TS	**83.78**	**85.85**	90.02	87.40	**65.68**	**65.91**	**72.31**	**66.64**

Table F.4: **Influence of TS on uniform vs long-tailed distribution.** Comparison of MoCo vs MoCo+TS on CIFAR10-Uniform and CIFAR-LT-imb100, one split. Evaluation metrics: kNN classifier, FS LP denotes few-shot linear probe, LT LP denotes long-tail linear probe.

F.4 INFLUENCE OF THE POSITIVE SAMPLES ON CONTRASTIVE LEARNING

In Section 9.1.2, we particularly focused on the impact of the *negative samples* on the learning dynamics under the contrastive objective, as they likely are the driving factor with respect to the semantic structure. In fact, we find that the positive samples should have an inverse relation with the temperature τ and thus cannot explain the observed learning dynamics, as we discuss in the following.

To understand the impact of the *positive samples*, first note their role in the loss (same as Equation (9.4)):

$$\mathcal{L}_{\mathrm{c}}^{i} = \log\left(1 + c_{ii} S_i\right) . \tag{F.1}$$

In particular, c_{ii} scales the entire sum $S_i = \sum_{j \neq i} \exp(-d_{ij})$. As such, encoding two augmentations of the same instance at a large distance is much more 'costly' for the model than encoding two different samples close to each other, as each and every summand S_i is amplified by the corresponding c_{ii}. As a result, the model will be biased to 'err on the safe side' and become invariant to the augmentations, which has been one of the main motivations for introducing augmentations in contrastive learning in the first place, cf. [TSP+20, CKNH20, CMM+20].

Consequently, the positive samples, of course, also influence the forming of clusters in the embedding space as they induce invariance with respect to augmentations. Note, however, that this does not contradict our analysis regarding the impact of negative samples, but rather corroborates it.

In particular, c_{ii} biases the model to become invariant to the applied augmentations for all values of τ; in fact, for small τ, this invariance is even emphasised as c_{ii} increases for small τ and the influence of the negatives is diminished. Hence, if the augmentations were the main factor in inducing semantic structure in the embedding space, τ should have the opposite effect of the one we and many others [WL21a, ZZP+22, ZWBG21] observe.

Thus, instead of inducing semantic structure on their own, we believe the positive samples to rather play a critical role in influencing which features the model can rely on for grouping samples in the embedding space; for a detailed discussion of this phenomenon, see also [CLL21].