Towards Holistic Machines:

From Visual Recognition To Question Answering About Real-World Images

A dissertation submitted towards the degree
Doctor of Engineering Science
(Dr.-Ing.)
of the Faculty of Mathematics and Computer Science
of Saarland University

by
Mateusz Malinowski, M.Sc.

Saarbrücken
06.2017

Defense

Day of Colloquium: 20.06.2017

Dean of the Faculty: Univ.-Prof. Dr. Frank-Olaf Shreyer

Examination Committee

Chair: Prof. Jens Dittrich

Reviewer, Advisor: Dr. Mario Fritz

Reviewer: Prof. Dr. Manfred Pinkal

Reviewer: Prof. Trevor Darrell, Ph.D.

Academic Assistant: Dr. Qianru Sun

Acknowledgments

First and foremost, I would like to thank Dr. Mario Fritz for giving me an excellent opportunity to work under his guidance at Max Planck for Informatics. His advice and scientific experience have helped me to shape my personality as a scientist, and grow as a researcher. His enthusiasm and willingness to cross the boundaries of Computer Vision have resulted in many great memories, also manifested by our projects. I would also like to thank Prof. Bernt Schiele, whose expertise and enormous energy have made this lab, with so many great researchers with such diverse experience, very special.

I am truly grateful for Prof. Trevor Darrell and Prof. Manfred Pinkal for serving as reviewers on the thesis commmitte. Their comments on my work has influenced me to work further on this exciting project. Special thanks to Prof. Jens Dittrich for being the chair of the committee and Dr. Qianru Sun for being a part of the committee as well.

I am grateful that I had an opportunity to work with two students, Sreyasi Nag Chowdhury and Ashkan Mokarian, on different aspects of my main endeavor on Visual Turing Test. Similarly, collaboration with Apratim Bhattacharyya on boundary extrapolation has made me interested in the 'intuitive physics'. I wish all the best in your scientific journey. Moreover, I am also grateful to my other three fantastic collaborators that I had a chance to work with: Dr. Zeynep Akata on her project on Zero-Shot Learning, Dr. Marcus Rohrbach whose advice helped me a lot on my most recent project, and Dr. Andreas Bulling with whom I was collaborating on the 'Collective Memories' project.

Many thanks to my office mate, Walon, whose cheerful attitude helped me during difficult times. He also supported me many times during our trips. I am truly happy you get an awesome position back at Taiwan!

I owe an enormous debt of gratitude to Jan Hosang, whose technical expertise made my work much easier. He has also helped me in many other ways during my stay at Germany.

Special thanks to Dr. Rodrigo Benenson, Dr. Micha Andriluka, Dr. Michael Stark, Dr. Björn Andres, and Dr. Peter Gehler. Their experience, research projects, and valuable comments not only have influenced me in many ways, but also helped me to shape my scientific journey. You guys are constant source of inspirations.

I would like to express my sincere gratitude to all (former and current) members of this lab, especially to: Maksim Lapin, Siyu Tang, Wenbin Li, Sabrina Hoppe, Julian Steil, Anna Khoreva, Anna Rohrbach, Leonid Pishchulin, Hosnieh Sattar, Seong Joon Oh, Abhishek Sharma, Yusuke Sugano, Mohamed Omran, Eldar Insafutdinov, Bojan Pepikj, Gaurav Sharma, Martin Simonovsky, Yongqin Xian, Yang He, Shanshan Zhang, Margret Keuper, Fabio Galasso. Many thanks to Connie Balzert for her excellent organizational work. You guys make this lab very special.

Without my friends, Adam Grycner, Przemek Grabowicz, Barbara Kościelecka, Asia Biega, Tomek Dudziak, Benjamin Roth, Kasia and Piotr Danilewscy, Krzysiek and Martyna Klimek, Marta Podgórska, Azim Dehghani Amirabad, Yulia Gryaditskaya, Monika Hosen, Basileios Anastasatos, Ashutosh Modi, and Lisette Noboa my life abroad would not be so exciting. I would like to especially thank to my close friend Konrad Jamrozik. He was always willing to support me in numerous ways. I wish your stay in USA will be a wonderful

experience. I would also like to thank Tomek Dudziak for lending his voice in the spotlight of our 'Ask Your Neurons' paper[1].

Spetial thanks to Konrad Jamrozik, Krzysztof Templin, Barbara Kościelecka, Seong Joon Oh, Jan Hosang, Mohamed Yahya, Kasia Danilewska, and Azim Dehghani Amirabad for proofreading this thesis. Your valuable comments helped me to shaped this thesis.

I would like to thank my parents, brothers and grandparents for their never-ending support in my endeavor. As a result of the sense of security that they have given to me I could have spent my time and energy on my PhD studies.

Last but not least, I want to thank my beloved girlfriend Joanna Pacia for her constant support and belief in me and my work. Without her help my stay abroad would not be possible. I am dedicating my thesis to her.

[1] https://www.youtube.com/watch?v=QZEwDcN8ehs

Abstract

Computer Vision has undergone major changes over the recent five years. Together with the advances on Deep Learning, and creation of large-volume datasets, the progress becomes particularly strong on the image classification task. Moreover, we also observe a succesful move from hand-designed to learnt features allowing to adapt to the task at hand. Therefore, we investigate if the performance of such architectures can also scale up to more complex tasks that require a more holistic approach to scene comprehension. This thesis has four main themes that have contributed to these advances in Computer Vision.

First, we extend Spatial Pyramid Matching, an integral part of most traditional visual recognition frameworks, by introducing a learning-based approach to discriminatively learn a spatial pooling layout from training data. Our results show a significant improvement over the original Spatial Pyramid Matching architecture, providing evidence that the hand-designed spatial division is indeed suboptimal.

Second, we have found a link between the pooling regions, and a computational model for spatial reasoning, which we applied to a text-to-image retrieval task. Interestingly, spatial pooling regions can also be related to spatial templates developed in psychological studies on human spatial reasoning. In this part of the thesis, we also explore a compositional neural architecture for the image retrieval task. An explicitly parameterization of the proposed method allows for spatial reasoning. This work is a precursor that has led us to the work on a Visual Turing Test.

The last two parts are about the Visual Turing Test, the task where a machine has to answer various questions about the content of images. In the third part, we have introduced, for the first time, the question answering task about real-world images. We have proposed DAQUAR, the first 'question answering about images' dataset together with the first method that handles the problem. Since the method relies on a semantic parser as well as a database of visual facts, we call the method the logic-based visual question answering architecture. In order to deal with uncertain visual inputs, we have proposed a bayesian extension to the semantic parser that runs over various possible interpretations of the visual scene. In this part, we have also introduced the first evaluation metric that embraces uncertainty in the word's meaning.

In the fourth part, we continue to work on the Visual Turing Test. Here, we have proposed the first end-to-end, jointly trained approach to the 'question answering about images' task. Since the method is a multimodal, Deep Learning method that combines a Recurrent Neural Network with a Convolutional Neural Network, we call the method a neural-based visual question answering architecture. In addition, we have collected an additional set of annotations and proposed two extensions of the evaluation metric to embrace uncertainty in various question and image interpretations.

In summary, this thesis contributes to the Computer Vision field in various ways: from the visual recognition, through image-to-test retrieval, to 'question answering about real-world images'. In most of the parts of the thesis, we argue for the jointly trained, neural-based methods where the representation of the input is optimized directly towards the end-goal.

Zusammenfassung

Computer Vision hat sich in den letzten fünf Jahren stark verändert. Zusammen mit den Fortschritten im Bereich Deep Learning und der Erstellung von umfangreichen Datensätzen wird der Fortschritt besonders im Bereich der Bildklassifizierungsaufgaben deutlich. Des Weiteren können wir einen erfolgreichen Übergang von manuell gestalteten zu erlernten Funktionen beobachten, der es ermöglicht, die jeweilige Aufgabe anzupassen. Daher untersuchen wir, ob die Leistung solcher Architekturen auch auf komplexere Aufgaben erweitert werden kann, die einen eher ganzheitlichen Ansatz an die Szenenerfassung verlangen. Diese These umfasst vier Hauptthemen, die zu diesen Fortschritten bei Computer Vision beigetragen haben. Zunächst erweitern wir Spatial Pyramid Matching, ein integraler Bestandteil der meisten konventionellen visuellen Erkennungsrahmen, durch Einführung eines lernbasierten Ansatzes, um auf unterschiedliche Weise ein räumliches Bündelungs-Layout von Schulungsdaten zu erlernen. Unsere Ergebnisse zeigen eine deutliche Verbesserung im Vergleich zur ursprünglichen Spatial Pyramid Matching-Architektur und liefern den Beweis, dass die manuell gestaltete räumliche Unterteilung in der Tat suboptimal ist.

Als Zweites haben wir eine Verbindung zwischen Bündelungsregionen und einem computerbasierten Modell für räumliche Argumentation gefunden, die wir auf eine Text-an-Bild-Wiederherstellungsaufgabe angewendet haben. Interessanterweise können räumliche Bündelungsregionen auch mit räumlichen Vorlagen in Bezug gesetzt werden, die in psychologischen Studien zur menschlichen räumlichen Argumentation entwickelt wurden. In diesem Teil der These untersuchen wir außerdem eine zusammensetzende neurale Architektur für die Bild-Wiederherstellungsaufgabe. Eine explizite Parametrisierung der vorgeschlagenen Methode ermöglicht die räumliche Argumentation. Diese Arbeit ist eine Vorstufe, die uns zur Arbeit an einem Visual Turing Test führte.

Die letzten beiden Teile betreffen den Visual Turing Test, die Aufgabe, bei der eine Maschine verschiedenste Fragen zum Inhalt von Bildern beantworten muss. Im dritten Teil haben wir zum ersten Mal die fragenbeantwortende Aufgabe zu echten Bildern eingeführt. Wir haben DAQUAR vorgeschlagen, den ersten „fragenbeantwortenden Datensatz zu Bildern“, gemeinsam mit der ersten Methode, die dieses Problem behandelt. Da diese Methode auf einem semantischen Parser sowie auf einer Datenbank an visuellen Fakten beruht, nennen wir diese Methode logikbasierte Fragenbeantwortungs-Architektur. Um mit den unbestimmten visuellen Eingaben umgehen zu können, haben wir eine Bayesian-Erweiterung für den semantischen Parser vorgeschlagen, die über verschiedenen Interpretationsmöglichkeiten der visuellen Szene ausgeführt wird. In diesem Teil haben wir auch die erste Bewertungsmetrik eingeführt, die die Unsicherheit in der Wortbedeutung behandelt.

Im vierten Teil arbeiten wir am Visual Turing Test weiter. Hier haben wir das erste End-to-End vorgeschlagen, ein gemeinsam geschulter Ansatz an die Aufgabe der „Fragenbeantwortung über Bilder“. Da diese Methode multimodal ist, die Deep Learning Methode, die ein Recurrent Neural Network mit einem Convolutional Neural Network kombiniert, nennen wir die Methode eine neural-basierte visuelle Fragenbeantwortungsarchitektur. Zusätzlich haben wir einen weiteren Satz an Anmerkungen gesammelt und zwei Erweiterungen der Auswertungsmetrik vorgeschlagen, um die Unsicherheit in machen Frage- und Bildinter-

pretationen zu behandeln.

Zusammenfassend trägt diese These zum Computer Vision-Bereich auf verschiedene Arten bei: von der visuellen Erkennung über Bild-an-Test-Wiederherstellung bis hin zur „Fragenbeantwortung zu realen Bildern“. Im Großteil dieser These plädieren wir für die gemeinsam geschulten, neural-basierten Methoden, bei denen die Darstellung der Eingabe direkt in Richtung des Endziels optimiert wird.

Contents

Chapter 1

Introduction

Contents

Holistic scene understanding is a long-standing goal of Artificial Intelligence (AI). From the early years of AI, the research community has attempted to develop machines that can interact with an environment through a natural language interface [Winograd 1971]. However, due to a series of failures in scaling up the work to real-world scenarios, which also shows how difficult the holistic comprehension is, the problem has been decomposed into several subproblems that gave a birth to different scientific disciplines such as Computer Vision, or Natural Language Understanding. All the aforementioned fields used to deal with different human faculties, and as such, they set different goals, and have developed different methods. A significant fraction of the work on Computer Vision or Natural Language Understanding is Feature Engineering – a discipline of designing a good representation of the input data. However, we also observe a successful move from such hand-designed features towards learning-based approaches that learn the features directly from data. So called Deep Learning, which employs deep neural networks, is often used as a synonym for the latter. In this thesis, we take a similar view, and show how to incorporate a learning-based approach to learn a spatial layout into a recognition framework that otherwise uses a spatial division set up a-priori and arbitrarily (Chapter 6).

Due to the recent advances in Deep Learning, researchers can not only use approaches to automatically learn features from data, but also successfully apply these techniques to achieve state-of-the-art results in the image classification [Krizhevsky et al. 2012; Szegedy et al. 2015; He et al. 2015], and object detection [Girshick et al. 2014; Ren et al. 2015b] tasks. Interestingly, Deep Learning has also unified, to some degree, Computer Vision with Natural Language Understanding, as many problems in each field can be tackled with methods similar in design. Typically, they give a vector representation of an image or a natural language sentence. This altogether has ignited interest of the research community in a multimodal setting, where vector representations of both modalities are combined. For instance, we benefit from such vector-based representations to build approaches to retrieve

images based on a textual query (Chapter 7) or to build approaches that answer questions about real-world images (Chapter 10).

At the same time there is also an interesest of the Computer Vision community in the general scene understanding, with different tasks set up as testbeds such as object detection, or semantic segmentation. However, as we are pushing towards a more holistic line of research, building large-volume datasets by collecting detailed annotations such as bounding boxes or per-pixel annotations becomes increasingly more problematic. Moreover, such 'visual understanding' should, arguably, also be to some extent agnostic to an internal representation of various methods. The majority of this thesis, introduces and discusses our alternative approach to 'understanding', where methods are evaluated based on their performance in answering questions about images. Due to the similarities of such a task to the famous Turing Test, we call it a Visual Turing Test (Chapter 8). Arguably, a good performance on such a question answering task is necessary for the human-quality holistic comprehension of the world.

To base upon our intuition that question answering about images requires a logical reasoning, and by drawing inspirations from the advances on the 'textual question answering' task, we build our first approach using a semantic parser together with an external dataset representing the visual world by storing uncertain visual facts (Chapter 9). We call such a method logic-based. However, such an approach, composed of many independent components, strongly relies on many design choices as well as the visual representation. This contrasts with the main idea of Deep Learning that emphasizes learning a representation directly from data. Therefore, considering that Deep Learning approaches have been shown successful in tackling many Computer Vision and Natural Language Understanding problems, we build our second approach using a combination of Recurrent Neural Networks with Convolutional Neural Networks. We call such a method neural-based (Chapter 10 and Chapter 11). This method, which significantly outperforms a logic-based one, can be jointly and end-to-end trained. Therefore it does require making fewer design choices, and can more effectively use the language to come up with answers to numerous questions.

To summarize, in this thesis, we tackle various problems from Visual Recognition, through Text-to-Image Retrieval, to Visual Turing Test. Considering that the more traditional approaches are often subject to many undesired issues, we argue for approaches that automatically build a representation of input data with minimal design decisions, jointly and end-to-end. First, each design decision introduces an error that cannot be easily corrected. Second, end-to-end, jointly trained architectures directly optimize towards the end-goal and therefore have a freedom to build an appropriate representation of the input data given the final task. Third, it is difficult to hand-design appropriate features in a multimodal domain. Fourth, such architectures are easier to scale up to new domains or to cover new scenarios as they do not require re-designing or extending the current features. Fifth, as we can see not only in this thesis but also in the whole field, such approaches exhibit competitive or even state-of-the-art performance compared with hand-designed ones. In addition to jointly trained and end-to-end architectures, we also argue for architectures that can reason about spatial relations. Ideally such methods should also be trained jointly and end-to-end. Finally, we argue for tackling more holistic tasks such as a Visual Turing Test for

the following reasons. First, owing to the progress on the individual disciplines, Computer Vision and Natural Language Understanding, we have strong tools to build a representation of sensory inputs at our disposal. This enables the research community to work on more holistic problems. Second, building holistic machines that understand sensory inputs, and can perform actions accordingly is a challenging, open problem. It is particularly interesting to see if Deep Learning can be extended to model higher cognition. Third, our variant of a Visual Turing Test does not depend on an internal representation of the input data as the performance is only measured on the final task. Fourth, arguably the Visual Turing Test does not require a high annotation effort in comparison with the image segmentation task, and at the same time is more focused than the image description task.

In the remainder of this chapter, we will revisit our contributions in the context of the whole thesis as well as the whole research field. Later on, we also briefly outline individual chapters of the thesis.

1.1 Contributions of the Thesis

This thesis contributes to Visual Recognition, Deep Learning, Natural Language Understanding, and Multimodal Learning in various ways. In this section, we have grouped our contributions, and put them in the relation with other work in the field. Note that contributions are also presented in individual chapters of the thesis.

Task The most prominent contribution of the thesis is our approach to determine a good holistic task. On one hand, we would like to have a task that tests visual scene understanding, however, in contrast to detection and segmentation tasks, with a more scalable annotation process, and one that is to some extend agnostic to an image representation (does neither require outputting bounding boxes nor per-pixel labels). On the other hand, in contrast to the image description task, we would like to have objective ways to automatically monitor the progress on the task via different evaluation metrics. With all these goals in mind, we have introduced a task where machines have to answer questions about real-world images, which we call a Visual Turing Test. In this task, a series of questions about the visual content of an image are used to test the 'understanding'. Therefore the overall performance is only measured based on provided answers, and does not need to evaluate an internal representation of various methods. In other words, we choose natural language as the final measure of 'understanding' irrespective of any internal specifics of a visual representation. Moreover, the task, arguably, encourages researchers to take a more holistic view as the whole chain of visual understanding, question understanding, understanding of human intentions, and deductive capabilities is needed to come up with an answer to the given questions about the image. Since our first proposal of the task that comes along with a dataset and a method [Malinowski and Fritz 2014a,b, 2015], many other research labs have followed up our work by proposing different datasets that model such a problem (Section 5.3.2). Chapter 8 gives a more detailed exposition on the task.

Datasets Datasets represent some parts of the real-world by abstracting away many nuances thereof. For the Visual Turing Test, also recently more commonly known as Visual Question Answering, we have proposed the first dataset for the task, which we call DAQUAR, consisting of question-answer-image triples [Malinowski and Fritz 2014a]. The dataset reflects challenges that machines need to deal with. Most prominently different question and scene interpretations, various naming conventions, ambiguities in spatial relations, reasoning about states, abstract references, and small objects. Since our introduction of DAQUAR, other datasets have also been proposed, including Visual Turing Test from Geman et al. [2015], VQA [Antol et al. 2015], Image QA [Ren et al. 2015a], FM-IQA [Gao et al. 2015], Visual Madlibs [Yu et al. 2015a], Visual7W [Zhu et al. 2016], Collective Memories [Chowdhury et al. 2016a], filling the blanks in captions for videos dataset [Zhu et al. 2015], MovieQA [Tapaswi et al. 2016], FVQA [Wang et al. 2016], KB-VQA [Wang et al. 2015], and SHAPES [Andreas et al. 2016b]. All the aforementioned datasets represent various aspects of the whole 'question answering about images' task. Chapter 8 and Chapter 9 introduce DAQUAR, while Section 5.3.2 enumerates, with a greater attention to details, related datasets that follow up our DAQUAR.

To answer a significant fraction of the questions in DAQUAR some sort of spatial resolution is needed. To study spatial relations in isolation, we have built a dataset of structured and compositional human queries [Malinowski and Fritz 2014c]. Structured queries have form (object, spatial relation, object), while compositional queries are more complex, e.g. (object, spatial relation, structured query). Chapter 7 and Appendix A introduce the dataset.

Methods for a Visual Turing Test Together with the task, and the dataset, we have introduced the first, logic-based, method that answers to questions about real-world indoor images [Malinowski and Fritz 2014a]. We build a database of visual facts, and use a semantic parser to transform questions into the formal representations. Such a representation is next executed on the database in order to conclude the answer to the question. Our next method, which we call a neural-based method, combines LSTM with CNN [Malinowski et al. 2015]. This is the first jointly trained, end-to-end architecture that generates an answer to the question. Both methods are presented in Chapter 9 and Chapter 10. Most of the subsequent models that have appeared in this domain are based on similar ideas shown in this thesis [Ren et al. 2015a; Gao et al. 2015; Antol et al. 2015; Yu et al. 2015a; Chowdhury et al. 2016a; Zhu et al. 2016, 2015; Ilievski et al. 2016; Yang et al. 2015; Xiong et al. 2016; Shih et al. 2016; Chen et al. 2015; Wu et al. 2016b,a; Wang et al. 2016, 2015]. Finally, Chapter 11 extends our work from Malinowski et al. [2015] to an analysis on VQA – the largest currently available dataset for the 'question answering about images' task. At the time of writing, to the best of our knowledge, the method presented in Chapter 11 is the best performing VQA architecture that uses a global, full-frame image representation [Malinowski et al. 2016].

Multimodal Recurrent Neural Networks Approaches that answers to questions about images condition on the question and the image in order to infer an answer. However, prior Recurrent Neural Networks have conditioned only on one modality. Therefore, we extend

LSTM – a particularly popular Recurrent Neural Network – to condition on the question and the image [Malinowski et al. 2015, 2016]. Moreover, our neural-based architectures are also capable of generating answer words. All the aforementioned architectures are shown in Chapters 10 and 11.

Semantic parser and probabilistic databases The Semantic Parser that we use in our logic-based approach to question answering is originally applied only to deterministic factual databases such as a knowledge base of geographical facts [Liang et al. 2013]. However, in order to work with the uncertain output of various visual analysis techniques, we have extended the semantic parser of Liang et al. [2013] to handle probabilistic databases containing (uncertain) visual facts [Malinowski and Fritz 2014a]. This is covered in Chapter 9.

Data-driven Recursive Neural Networks To ground spatial prepositions in structured queries, as well as to explicitly exploit the compositional structure in language, we have developed a recursive architecture, which topology is determined by the output of a syntactic parser. This architecture, used in Malinowski and Fritz [2014c], differs from Socher et al. [2011] by using different parameterization of the network that explicitly encodes spatial relations, and using of the syntactic parser. It is also similar to neural modules of Andreas et al. [2016b], but mostly focuses on spatial relations. Appendix A presents a more detailed exposition of the architecture.

Smooth learnable pooling regions In this thesis, we extend Spatial Pyramid Matching, a traditional recognition architecture, with a learning-based approach to learn a spatial layout used for pooling [Malinowski and Fritz 2013b,a]. We first generalize the pooling operator over large spatial regions by a suitable parameterization, and next we jointly train the pooling layout together with a classifier. Finally, we have shown the importance of smooth transitions between the pooling regions. This is implemented with a total variation regularization term. Interestingly, the learnt spatial layout (shown in Table 6.2) together with the objective function (shown in Equation 6.5) shows some similarities with attention masks, which are common in neural approaches to image description or question answering tasks [Xu et al. 2015; Xu and Saenko 2015]. However, once learnt the pooling regions are fixed. Chapter 6 covers this topic.

Differentiable generalization of the pooling operation Malinowski and Fritz [2013b,a] generalize a pooling operator with a spatial parameterization. Moreover, our generalization is differentiable, and therefore can be jointly trained together with a classifier via backpropagation. Interestingly, we are also observing an interest of the Deep Learning community in determining a suitable generalization of the pooling operator for Convolutional Neural Networks, e.g. by sampling weights of the pooling regions or by mixing max-pooling with the average-pooling [Zeiler and Fergus 2013; Lee et al. 2016]. The generalization is shown in greater detail in Chapter 6.

Spatial reasoning Spatial reasoning plays an important role in holistic tasks such as Visual Turing Test or text-to-image retrieval task [Malinowski and Fritz 2014c,a,b, 2015;

Chowdhury et al. 2016a]. However, spatial reasoning is challenging, mostly because at least two objects are involved, the notion of the spatial relation is unclear, and the existence of ambiguities [Malinowski and Fritz 2014a,b]. Therefore, we have built a dataset, where spatial relations can be studied in isolation [Malinowski and Fritz 2014c], as well as a neural-based architecture with an explicit spatial reasoning derived based only on training data. All are covered in Chapters 7, 8, and Appendix A.

Pooling in spatial reasoning Inspired by studies in psychology [Logan and Sadler 1996], we have shown a link between spatial templates proposed in the psychological studies [Logan and Sadler 1996] and learnable pooling regions of Malinowski and Fritz [2013a]. Next, we use the ideas of the pooling regions [Malinowski and Fritz 2013a] to build a data-driven approach to spatially reason about the objects in the image via the spatial templates. Finally, we have successfully applied this technique to the text-to-image retrieval task. This is covered in Chapter 7.

Learning representations The thesis contributes to the Learning Representation field, where the goal is to devise automatic approaches to learn a representation of the input. This contrasts with Feature Engineering, where the representation (features) are mostly hand-designed. Our work contributes to the field via devising methods that learn a multimodal representation that combines language with vision [Malinowski et al. 2015, 2016], methods that learn a spatial layout of the image for the image classification task [Malinowski and Fritz 2013b,a], and methods that learn to spatially reason about the objects in the scene [Malinowski and Fritz 2014c]. These contributions are covered in greater detail in Chapters 6, 7, 10, 11, and Appendix A.

Evaluation metrics Although, by constraining a set of possible answers we have limited the complexity of the output space in the DAQUAR dataset, some ambiguities still remain. Such ambiguities need to be taken into account while evaluating methods on the dataset. Our first evaluation metric, WUPS, is a generalization of the traditional Accuracy measure that works with sets of answers, and takes care of word-level ambiguities such as various names given to the same 'thing' [Malinowski and Fritz 2014a,b]. To deal with various visual and question interpretations, we have next proposed two Consensus Measures: Min Consensus, and Average Consensus. Both metrics provide different insights into the task; Min Consensus desirably gives higher scores to human answers, while Average Consensus can down-weight or even filter out controversial answers. The latter also partitions the dataset according to the human agreement level. All the aforementioned metrics are important contributions to Visual Turing Test. WUPS as well as its extension, Consensus Measures, are presented in Chapter 9 and Chapter 10 respectively.

1.2 Contributions to Other Projects

While making contributions along the main lines of the thesis, the author of this dissertation has also been actively engaged in other research projects by either advising, consulting or

helping in programming. In the following, the additional contributions are enlisted together with short descriptions of the projects.

1.2.1 Intuitive Physics

To study an intuitive understanding of physics by neural architectures, we have set up a task of predicting future frames based on an input video. Since a direct prediction of the pixel values is a complex task, and is arguably an unnecessary nuisance in understanding of physics, we have decided to reduce the complexity in the output space by setting up the goal of predicting future boundaries. In addition to the task definition, we have developed three neural-based architectures to handle the problem. Our experiments show that the proposed architectures are able to predict boundaries of many future frames without a noticeable degradation in the quality of predictions. Moreover, our results, especially the ones conducted on two billard datasets, show the models exhibit some understanding of physics. More details can be found in Bhattacharyya et al. [2016a], and Bhattacharyya et al. [2016b].

The author of this dissertation helped in developing a Convolutional RNN, and guided the collaborators on using automatic differentiation tools used in the further development of neural-based architectures.

1.2.2 Zero-Shot Learning

Zero-shot learning studies a problem of making predictions in the scenario where no training data of test classes are available. In the concrete scenario that we are interested in, we train a model to transfer an auxiliary information from known to unknown classes in order to recognize fine-grained, a-priori unseen, birds categories. Ideally, this extra information comes from readily available textual resources. In this work, we have extended the prior work on zero-shot learning to incorporate a richer source of information in the form of textual or visual parts, and we trained a deep architecture to recognize unknown images. Our results improves over the prior state-of-the-art on this topic. More details can be found in Akata et al. [2016].

The author of this dissertation has proposed the idea of using a richer, part-based, input representation. He has also helped in extending the retrieval framework of Karpathy et al. [2014] to this class of problems.

1.2.3 Visual Turing Test

Chapters 8, 9, 10, and 11 contain in-depth exposition to the Visual Turing Test that has been pioneered by the author of this dissertation together with the close collaboration with his supervisor Dr. Mario Fritz, and later widely followed up by the research community (often under an alternative name – Visual Question Answering). Concisely, the Visual Turing Test benchmarks scene and language understanding capabilities of Machine Learning architectures via a series of natural language questions about the content of natural images. Such the challenge requires a holistic approach towards the scene understanding. In addition

to the main line of research outlined in this dissertation in the aforementioned chapters, the author of this dissertation has helped Dr. Mario Fritz in advising two students on this topic.

The wide body of work on Visual Turing Test considers static scenario where questions and images are given. In contrast, due to a widespread of mobile and wearable computing platforms, in Chowdhury et al. [2016a] we envision a situated scenario where natural language questions are contextualized in a dynamic, ever-changing environment through GPS coordinates and timestamps. That is, an user asks questions 'what is in front of me' or 'how did this place look like in December' and expects from the system to retrieve appropriate media content such as images based on the user's geolocation and the question. In this work, we have evaluated our extension of the logic-based question answering architecture (Chapter 9) on a newly collected dataset of images, GPS coordinates, and natural language questions. Through our human studies, we show that the presented architecture can cope well in this situated and dynamic scenario. The author of this thesis came up with the problem definition where images are treated as 'answers', and helped the student to become comfortable with the suitable framework.

In Mokarian Forooshani et al. [2016], we have extended the neural-based approach to question answering about images (Chapter 10 and Chapter 11) to work with a richer image representation in the form of object proposals. Quite counterintuitively, we have found that a high number of highly overlapping object proposals lead to better results than a small number of precisely localized ones. In our interpretation of such results, we conclude that the former approach leads to a multi-part and multi-scale rich image representation that helps in automatic reasoning. Moreover, we have also taken advantage of Visual Madlibs [Yu et al. 2015b] formulation, where the architecture is supposed to fill the blank in a sentence describing a natural image from a set of pre-defined candidates, and trained the neural-based architecture by maximizing similarities between latent representations of the multimodal input and candidate answers. The author of this thesis helped the student to become comfortable with the suitable framework.

1.3 Outline of the Thesis

In this section, we summarize and relate to each other different chapters of the thesis. Chapters 1, 2, 3, 4, 5, and Chapter 12 introduce an unpublished material, while the remaining chapters have been shown in various conferences, technical reports (arXiv), or workshops. Mateusz Malinowski is the lead author of all the papers contributing to the thesis. His supervisor, Dr. Mario Fritz, is a co-author of all the papers presented in the thesis. Dr. Marcus Rohrbach has contributed to two papers presented in the thesis in Chapter 10 and Chapter 11. He advised us on the LSTM implementation in Caffe as well as he helped in editing both papers.

Chapter 1: Introduction This chapter gives a brief introduction to the whole thesis. It also introduces various research directions that are investigated in the thesis. Next, it presents how the thesis contributes to the ongoing research in Computer Vision, Natural Language Understanding, and Deep Learning. Finally, this chapter outlines individual chapters and

appendices of this thesis.

Chapter 2: From Visual Recognition Towards Holistic Machines The remainder of the thesis starts with giving a brief historical context on two disciplines: Computer Vision and Natural Language Understanding. It shows that together with large-volume datasets as well as the development of Deep Learning methods, both communities have developed methods that greatly perform on many traditional tasks in each field of study. Interestingly, similar methods can be used to tackle many problems in both disciplines. Considering this as a motivation, the chapter argues for seeking more complex, holistic tasks that unites both disciplines together. It also argues for a development of holistic methods that can handle such tasks. Finally, the chapter introduces the 'question answering about real-world images' task (also called Visual Turing Test, Visual Turing Challenge, or Visual Question Answering), as an example of a holistic problem.

Chapter 3: Background: Visual Recognition The subsequent chapter provides a background knowledge about the Computer Vision methods used in this thesis. It formalizes the machine recognition, where an informal introduction has been given in From Visual Recognition Towards Holistic Machines. It shows Spatial Pyramid Matching that used to be the leading recognition architecture, and is further extended in Learning Smooth Pooling Regions for Visual Recognition. This chapter also shows Convolutional Neural Networks (CNNs) that have replaced Spatial Pyramid Matching for the recognition task. What is more important, Convolutional Neural Networks can be used to extract features for other tasks. Later on, we use CNNs features to build our neural-based approaches to handle question answering about images. Finally, the chapter briefly describes the most recent CNNs architectures.

Chapter 4: Background: Natural Language Understanding Similarly to Background: Visual Recognition, this chapter provides a background knowledge about the Natural Language Understanding methods used in the thesis. The chapter first describes a semantic parser. Next, it introduces a neural-based approach to model the language. We use a semantic parser in our first, logic-based approach towards a Visual Turing Test. A neural-based representation of the language has been used in our second, neural-based, approach to answer questions about images.

Chapter 5: Related Work The next chapter extends the historical context given in the previous chapters by providing more references. It also explicitly depicts a progress in methods that handle or datasets that represent the Visual Turing Test. Finally, it relates each chapter corresponding to our publication with the whole subfield - prior and subsequent work.

Chapter 6: Learning Smooth Pooling Regions for Visual Recognition This chapter is based on our two publications, Malinowski and Fritz [2013b] and Malinowski and Fritz [2013a], presented in the *International Conference on Learning Representations (ICLR)*

workshop, and *British Machine Vision Conference (BMVC)* respectively. Our work, presented in this chapter, extends the Spatial Pyramid Matching architecture (Chapter 3 has a more detailed exposition of Spatial Pyramid Matching) by allowing a more holistic, joint training of a spatial layout together with a classifier via backpropagation.

Chapter 7: A Pooling Approach to Modelling Spatial Relations for Image Retrieval and Annotation This chapter brings our ideas on learning spatial pooling regions from Chapter 6 into an image-to-text retrieval scenario with the goal of retrieving an image that matches the provided textual query. We show that such learnable pooling regions can subsequently be used to reason about spatial relations. Scene understating, natural language understanding, and spatial relations grounding arguably play important roles in holistic tasks. This chapter is based on our Technical Report available on *arXiv* [Malinowski and Fritz 2014c].

Chapter 8: Towards a Visual Turing Challenge The first exposition to a novel holistic task – a Visual Turing Test – that studies if machines can answer questions about images, is presented in this chapter. This chapter enumerates challenges that holistic machines have to handle, shows concrete challenges that are present in DAQUAR - the first Visual Turing Test dataset - and finally discusses a problem of measuring the performance of holistic architectures. Moreover, question answering about images can also be seen as a generalization of the image-to-text retrieval tasks, for instance the one presented in Chapter 7. This chapter is based on our two workshop expositions: Malinowski and Fritz [2014b] and Malinowski and Fritz [2015]. The former was presented at *NIPS Workshop on Learning Semantics*, while the latter at *AAAI: Beyond the Turing Test.*

Chapter 9: A Multi-world Approach to Question Answering about Real-World Scenes based on Uncertain Input The first dataset representing Visual Turing Test, called DAQUAR, is introduced in this chapter. Together with the dataset, we have also introduced a holistic, logic-based approach to handle the task. It is the first method that answers to questions about real-world images. Finally, we also present a new metric that extends standard Accuracy metric to handle world-level ambiguities. Similarly to Chapter 7, spatial relations consist of a significant part of the task. The work presented here is a concretization of our abstract ideas shown in Chapter 8. This chapter is based on our publication [Malinowski and Fritz 2014a] presented at *Neural Information Processing Systems (NIPS).*

Chapter 10: Ask Your Neurons: A Neural-based Approach to Answering Questions about Images The architecture shown in Chapter 9 has a few limitations. For instance, it assumes a hand-designed representation of an image, and cannot be jointly trained. To alleviate such limitations, we introduce in this chapter a neural-based approach to handle question answering about images. It is our second Visual Turing Test architecture, which also liberates us from many design decisions and allows for a holistic, joint training. Moreover, we also provide further

insights to the problem of question answering about images, and show two extensions of the previously introduced, in Chapter 9, metric. This work is based on our publication [Malinowski et al. 2015] presented at *International Conference on Computer Vision (ICCV)*.

Chapter 11: Ask Your Neurons: A Deeper Analysis The analysis shown in Chapter 10 has been subsequently extended in this chapter. Here, we conceptually divide the architecture from Chapter 10 into a few different modules, and next analyze influences of different modules on the task by replacing them. Our analysis shows benefits of using LSTM over BOW as well as stronger recognition architectures. Most of the analysis is performed on the VQA dataset. This chapter is based on our submission to *International Journal on Computer Vision (IJCV)*.

Chapter 12: Conclusions and Future Perspectives The last chapter concludes the whole thesis, and presents possible future directions in machine recognition, text-to-image retrieval, spatial relations, and question answering about images. In other words, the chapter 'holistically' binds different chapters together into a concise summary and possible further extensions.

Appendix A: DDCNA: Data-Driven Compositional Neural Architecture for Image Retrieval based on Compositional Queries In Chapter 7 we either augment the Deep Fragments Embeddings of Karpathy et al. [2014] with spatial templates, or use our compositional data-driven neural architecture. The latter is presented in greater detail in this appendix. The topology of this architecture is formed from textual queries, and therefore it is instance-dependent. Training is possible as the parameters are shared across many different input-induced architectures. The network explicitly uses rules of compositionality, and does a spatial reasoning.

Appendix B: Visual FactNet In this appendix, we show initial results with Visual FactNet– our novel approach to question answering about images. This architecture blends a neural-based approach to represent the language of Malinowski et al. [2015], which we present in Chapter 10, with an explicit visual representation of Malinowski and Fritz [2014a], which we present in Chapter 9. This appendix also offers a more fine-grained study of the performance of our initial neural-based architecture from Chapter 10.

Appendix C: Tutorial on Answering Questions about Images with Deep Learning This appendix shows our tutorial on the 'question answering about real-world images' task [Malinowski and Fritz 2016]. We made an effort to make it accessible to a broader audience. In particular, we do not assume any prior knowledge on Computer Vision, nor Natural Language Understanding. The tutorial was shown during the 2nd Summer School on Integrating Vision and Language: Deep Learning, in Malta, in 2016.

Although it is recommended to read the thesis consequently starting from Chapter 1, there is a possibility to read chapters in a different order. To avoid confusion, the dependencies

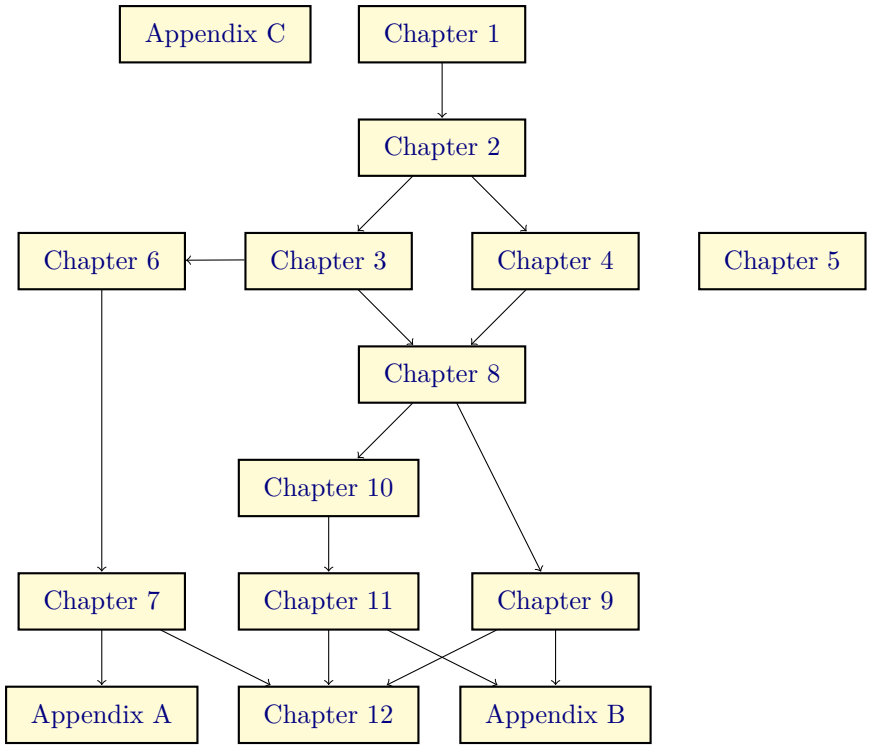


Figure 1.1: Graph depicting dependencies between different chapters.

between chapters are included in Figure 1.1. Appendix C and Chapter 5 can be read independently from other chapters.

Chapter 2

From Visual Recognition Towards Holistic Machines

Contents

Visual and linguistic information is ubiquitous in the human world. Humans can not only recognize objects, understand words, or combine the words into meaningful phrases or sentences, but also relate linguistic information to their visual counterparts, understand abstract concepts, or understand intentions hidden in the utterances of others. Clearly, human beings operate at different linguistic levels from syntax to pragmatics, and successfully combine such cues with a visual content. With such abilities, humans can quickly communicate with each other, act on the external world, and collaborate to perform more complex, holistic tasks. Similar behaviour is also expected from intelligent machines. Unfortunately, what is natural to humans is often difficult for machines, therefore such a human-quality understanding of the world is nowadays rather a part of science-fiction stories than reality.

On the other hand, we observe a tremendous progress in machine recognition (Section 2.2, Chapter 3), and natural language understanding (Section 2.3, Chapter 4). Many contemporary approaches to vision or language are based on Deep Learning, and therefore they learn from training examples. With large-volume datasets (Section 2.1), such leading 'data-hungry' learnable architectures have shown impressive results on the image classification task, achieving a performance comparable to the human-quality recognition under a

constrained setting [Russakovsky et al. 2014]. Such a few ingredients – large-volume datasets, hierarchical architectures, joint and end-to-end training – have contributed to a successful move from hand-designed features to learnt ones. In this thesis, we adopt the same philosophy to develop architectures that jointly learn spatial pooling regions with a classifier (Chapter 6) [Malinowski and Fritz 2013b,a], are trained to retrieve images based on structured queries where a spatial reasoning plays an important role (Chapter 7, Appendix A) [Malinowski and Fritz 2014c], and can answer to questions about real-world images (Chapter 10, Chapter 11, Appendix B, Appendix C) [Malinowski et al. 2015, 2016; Malinowski and Fritz 2016].

The progress that has been made in Computer Vision and Natural Language Understanding sparks questions if similar methods can also be successfully trained on more holistic tasks that require among other things a scene understanding from a visual stimulus (e.g. object and action recognitions or how the things are spatially related to each other), natural language understanding (e.g. lexical understanding as well as relations between words that form a meaning of sentences), and human pragmatics (e.g. dealing with subjectivity, or being equipped with common sense). To study such problems, we propose a task that measures how well machines answer to questions about real-world indoor images [Malinowski and Fritz 2014b, 2015, 2014a] – a behavioral and holistic test that draws inspirations from the famous Turing Test [Turing 1950], and which we call a Visual Turing Test – together with two seemingly antagonistic approaches to handle the task: logic and neural based ones presented in Chapter 9 and Chapter 10 respectively [Malinowski and Fritz 2014a; Malinowski et al. 2015]. Considering that spatial reasoning is an important part of the human life, which is also reflected in our dataset implementing the Visual Turing Test, we also argue for architectures with a spatial component, or even to study such a problem in a more controlled setting (Chapter 6, Chapter 7) [Malinowski and Fritz 2013a, 2014c].

To sum up, we hypothesize that large-volume datasets, hierarchical, jointly and end-to-end trained architectures that can reason spatially and are able to answer to questions about real-world images form necessary ingredients to develop holistic machines that can achieve a human-quality understanding of the world. In the remainder of this chapter, we discuss some individual components of a hypothetical holistic machine that combines vision with language.

2.1 Large Volume Datasets

Recent progress in mobile sensor technologies have allowed consumers to readily record the visual world in the form of images or short videos with a relatively high resolution and low cost. In the nearest future, owing to such projects as Kinect[1] or Tango[2], we may expect an appearance of more data that capture 3D world, which will lead to even finer representation of the visual world. At the same time internet sharing platforms become ubiquities. Internet users share their pictures, videos, knowledge, stories, or ideas through Flickr[3], YouTube[4], web-based encyclopedias, blogs, social networks, or other creative internet communities.

[1] https://en.wikipedia.org/wiki/Kinect
[2] https://en.wikipedia.org/wiki/Project_Tango
[3] https://www.flickr.com
[4] https://www.youtube.com

This altogether leads to a large volume of readily accessible, high quality visual or textual data. Therefore, the building of large-volume datasets for many tasks involving vision or language cannot be seen as a serious bottleneck anymore, and the advent of such large datasets created for the research purpose is inevitable. For instance, the ImageNet dataset is a particularly popular large-volume recognition dataset created to train and test image classification methods [Russakovsky et al. 2014]. In the remainder of this section, we first discuss some ramifications of the existence of large-volume datasets. Subsequently, we discuss two aspects of a scalability. Both aspects are tightly connected with large-volume datasets, and are important in the development of learnable architectures.

The existence of large-volume recognition datasets have led to an increased interest in data-driven, and 'data-hungry' methods that work with minimal modeling assumption and instead let the data to guide the inference [Hays and Efros 2007]. Especially, Deep Learning methods become recently very popular due to their apparent scalability (we revisit the scalability later in this section). For instance, the most successful recent Deep Learning approaches to image classification have achieved a near human-quality performance on the aforementioned ImageNet dataset [Russakovsky et al. 2014]. Chapter 3 covers this topic in greater detail. Nonetheless, one can argue that building intelligent machines that can exploit large amount of data is therefore both inevitable and actually human-like as human beings have also an access to abundant and diverse data.

As already mentioned in this section, computational and learnable scalability should be sought while developing new learning-based methods. On one hand, it should be feasible to train models on large amount of data preferably in an online manner, and to quickly perform an inference at the test time. On the other hand, the models should have high enough capacity to leverage an increase in the volume of data. Although there is often a trade-off between both aspects of scalability, it seems the recent Deep Learning approaches to image classification can be quickly run on modern GPUs, as well as they can take advantage of a large amount of data available during the training. For instance, Convolutional Neural Networks can reportedly perform an inference in about 1ms per image while achieving the state-of-the-art performance on the image classification task[5]. Later on in this thesis, the apparent 'sweet spot' of Deep Learning in terms of scalability has motivated us to develop neural-based approaches that answer to questions about real-world images (Chapters 10 and 11).

2.1.1 Concluding Remarks

Training machines on large-volume datasets has enabled methods to recognize objects in images at the accuracy that has not seen before (Section 2.2 and Chapter 3). However, the question whether we can further use visual or textual data to build and benchmark machines with a hypothetical human-quality world comprehension still remains open (Section 2.4). Nonetheless, in this dissertation, we attempt to partially answer to this question by introducing the first dataset along with the first approaches to answer questions about real-world images – a task that arguably is necessary to acquire a human-quality holistic

[5] http://caffe.berkeleyvision.org/performance_hardware.html

comprehension.

2.2 Visual Recognition

Visual Recognition is the task to recognize an object in the given image by machines. First approaches to recognition have reduced the visual reality to a geometrical world by fitting geometrical primitives to an image [Roberts 1963; Agin and Binford 1976; Mundy 2006], or by representing an object by its (still geometrical) components [Biederman 1987]. Later on, the research community shifted towards appearance-based methods, in which an appearance of an investigated object (often represented by such cues as a color or pose) is matched with appearances of already known objects [Murase and Nayar 1995; Swain and Ballard 1991]; or feature-based methods, in which local representations (e.g. histograms of oriented gradients extracted from a small patch) are matched with the representations of the already known (training) images [Lowe 2004; Dalal and Triggs 2005]. Together with larger recognition datasets (Section 2.1), recognition methods began to exploit machine learning techniques to successively build global representations from local ones [Csurka et al. 2004; Lazebnik et al. 2006; Yang et al. 2009]. Most recently, data-driven, neural-based, jointly optimized, multi-layer methods have become dominant in the recognition task (Chapter 3) [Krizhevsky et al. 2012]. In the remainder of this section, we first illustrate some challenges that recognition architectures have to deal with. Next, we discuss the role of an image representation in Computer Vision. Finally, we briefly describe three popular recognition tasks and discuss some constraints they impose on the image representation.

Although natural to humans, recognition is challenging for machines for various reasons. Why is it so difficult? There is a large variability in object categories. For example, instances of the same category can vary in shapes or appearances (e.g. cat breeds), which makes it difficult to precisely describe what the whole class is. Even further, we can observe a large variability in different images of the same instance due to occlusions, self-occlusions, changes in a scale or lighting conditions, and other complex rigid or non-rigid transformations. Since there are myriad of ways that pixels belonging to the same object can change, working directly on the pixel level seems to be not particularly fertile. Therefore the research community has to develop a higher abstraction that represents an object. This would explain various directions in developing approaches to recognition, which some of them we already mentioned in this section, and the importance of an image representation in general.

In principle, the community seeks a representation that not only preserves a notion of the category under all the 'interesting' category-preserving transformations[6] like the aforementioned changes in lighting conditions, but also a representation that makes it easy to discriminate instances of different categories. Let us consider two extremal examples. The pixel-level representation, on one hand, is very discriminative, i.e. it is unlikely that two images of different people have the same pixels. On the other hand, it fails as an invariant representation, e.g. different lighting conditions often result in color changes of the pixels.

[6] Let $c(o)$ denote a category of an object o. A transformation $\mathcal{T}$ (e.g. translation) that acts on the object o preserves the category if $c(o) = c(\mathcal{T}(o))$. Note that, at test time category-to-object assignment is unknown. Moreover, the image of $T(o)$ is often much different than the image of o. This makes the recognition difficult.

Alternatively, a mapping an image into a constant number would make the representation perfectly invariant under (all) transformations, and at the same time completely useless due to the lack of a discriminative power. Such a trade-off between invariances and a discriminative power has to be taken into account when designing a representation[7].

In addition to all the aforementioned issues with a representation, one can also argue that the list of all the 'interesting' category-preserving transformations is actually unknown. Therefore the Deep Learning community opposes to hand-designing such a representation, and instead they call for learnt representations based on the training data. We adopt a similar viewpoint in most of the chapters of this thesis.

Now, we briefly describe three popular Visual Recognition tasks: image classification, detection, and segmentation. Next, we discuss some constraints that different tasks impose on the algorithms that handle them. All the aforementioned tasks are also briefly depicted in Figure 2.1. All of them are encountered in different parts of this thesis.

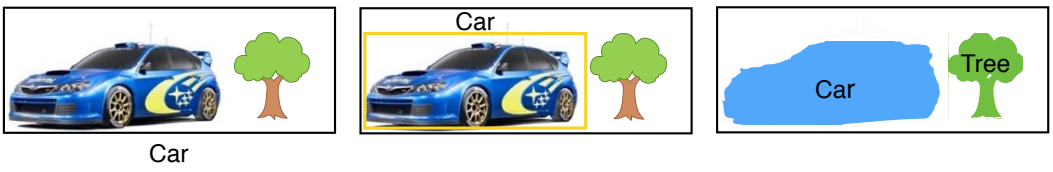


Figure 2.1: From left to right: image classification, detection, and segmentation tasks. The image classification is about assigning a category to an image, detection classifies and localizes an object, while segmentation asks for a more detailed scene representation.

Image classification Often, and by default in this thesis, recognition is set up as an image classification. The goal of the image classification is to categorize a given image into one out of K pre-defined categories. Low-volume datasets, which are previously used, tend to use mutually exclusive categories, but with the advent of larger datasets with a larger number of categories the semantic boundaries that distinguish different categories become more fuzzy. Nonetheless, most approaches handle the image classification task by training a K-out classifier on top of (hand-designed, or jointly learnt) features. Often images in the image recognition datasets have a single, main theme that has to be inferred (classified into) by a machine.

Detection As objects appear at different scales and locations in the image, the object detection task deals with localization of an object as well as its classification into a category. Pascal VOC dataset serves as a standard benchmark for detection [Everingham et al. 2015]. Similar to the image classification task, detection also "has undergone a seismic shift"[8] where traditional sliding window-based approaches with parts-based models and hand-designed features have been replaced by object proposals and learnt features [Girshick et al. 2014].

[7] Let $I(o)$ be an image of an object o. Ideally, we would like to have a representation R such that $R(I(o)) = R(I(\mathcal{T}(o))$ if $\mathcal{T}$ is a category-preserving transformation, and $R(I(o)) \neq R(I(\mathcal{T}(o)))$ otherwise.

[8] https://pdollar.wordpress.com/2013/12/10/a-seismic-shift-in-object-detection/

Segmentation We can aim at even richer representation of an image by grouping pixels together according to their semantics. For instance, pixels that correspond to human beings are assigned to a different class than pixels that correspond to furnitures. Furthermore, we can distinguish a class-based and instance-based segmentation. The former distinguishes only classes while the latter groups pixels that belong to the same instance of a class. For example, pixels belonging to one person are grouped separately from pixels belonging to another person. Methods handling semantic segmentation include: clustering of local image features or cutting edges in a graph describing affinities between pairs of pixels. Most recently, deep convolutional-deconvolutional architectures [Noh et al. 2015a; Badrinarayanan et al. 2015] have gained an attention in the research community. Semantic segmentation, arguably, demands a more holistic image understanding than detection or image classification.

Figure 2.1 depicts all the aforementioned tasks: image classification, detection, and segmentation. Every such a task imposes different constraints on a representation of images. The image classification cares only about final classes, and therefore is the most liberal in terms of the choice of the image representation. The object detection demands from methods to output some geometrical primitive describing an object. Most often it is a bounding box containing an object of interest. The semantic segmentation asks for a quite rich visual representation by the means of per-pixel classes. Therefore some methods that work quite well for the image classification task such as Bag-Of-Visual-Words [Csurka et al. 2004; Lazebnik et al. 2006], are not that commonly used in other tasks such as the semantic segmentation where the localized information should be preserved.

2.2.1 Concluding Remarks

Models trained to recognize objects also play an important role in other, also holistic, tasks. Therefore, every bit of progress on the recognition task translates into better visual features used by more complex architectures and simultaneous advances on the tasks they handle [Ren et al. 2015b; Venugopalan et al. 2015a; Malinowski et al. 2016]. As we see in this section, deriving a good image representation plays an important role in Visual Recognition. Such a representation can be either hand-designed or learnt from data. The latter becomes recently very popular in the research community due to its strong performance and less effort that is put on designing various architectures. This viewpoint is also dominant in this thesis. For instance, in Chapter 6 and Chapter 7, we argue for a learning-based approach to learn a spatial representation, while in Chapter 10 and Chapter 11, we use Deep Learning to build approaches that answer to questions about real-world images. Finally, different tasks put some constraints on the image representation. In this thesis, we argue that the 'question answering about real-world images' task is on one hand as liberal as image classification regarding the imposed constraints on the image representation. On the other hand, it requires a quite rich understanding of the visual world that, arguably, can be compared to the image segmentation task. We revisit this argument in Chapter 8 and Chapter 9.

2.3 Natural Language Understanding

In this section, we discuss the third pillar of the hypothetical holistic machine that combines language with vision. Natural Language Understanding by machines deals with comprehension of a textual, human-readable input[9]. Similar to Visual Recognition presented in Section 2.2, understanding of language is a highly non-trivial task. To be able to understand language, machines have to deal with ambiguities, incompleteness, and vagueness of utterances. In the remainder of this section, we discuss some causes of ambiguities that, arguably, are introduced by breaking down the field into separated components, and therefore we argue for a holistic treatment of the language understanding. Next, we discuss two different approaches to represent the meaning. We revisit them later in Chapter 4, Chapter 9, and Chapter 10.

Syntax, semantics, and pragmatics are the three major branches of linguistics. Syntax allows to decompose a sentence into simpler units according to some grammatical rules. Semantics is about deriving a meaning $[\![s]\!]$ of a sentence s. Informally, it is about transforming strings into 'things'. Pragmatics is about finding real intentions of a speaker in a broader context. However, to truly understand language, all the branches should interplay with each other. Otherwise, many ambiguities may arise. Let us first consider the following sentence "He eats a meat with a fork". From a syntactical point of view, the phrase "with a fork" binds not only with "He eats" meaning the fork is used as a tool to eat the meal, but also with "a meat" meaning the fork is a part of the meal. Let us consider now another example with the two following sentences: "The trophy would not fit in the brown suitcase because it was too big" and "The trophy would not fit in the brown suitcase because it was too small" [Winograd 1972; Levesque 2011]. Humans can easily resolve the anaphoric references by pointing the following denotations $[\![\text{it}]\!]$ = 'trophy' in the first sentence, and $[\![\text{it}]\!]$ = 'suitcase' in the second sentence. However, these sort of ambiguities cannot be resolved based on words in isolation, and possessing some sort of a holistic understanding of the world is required. Arguably, such the interplay is also necessary for the Visual Turing Test, introduced in Chapter 8.

2.3.1 Symbolic Representation of the Meaning

In this thesis, we build two different approaches to the 'question answering about real-world images' task based on two approaches to represent the meaning. In the following, we present the first approach, which we call a symbolic or logic based. This approach forms a foundation to our method presented in Chapter 9. We first illustrate a hypothetical language of thoughts, a concept that exists in linguistics and philosophy of mind. Next, we briefly describe a computational tool that apparently has many ideas in common with the aforementioned linguistic concept.

Mentalese Mentalese is a hypothetical language of thoughts with its own, compositional syntax, where complex thoughts are built from simpler thoughts [Fodor 1975]. It is often seen as a formal representation of thoughts and, from a philosophy of mind's point of view,

[9] https://en.wikipedia.org/wiki/Natural_language_understanding

it is hypothesized that humans 'think' in Mentalese. One compelling trait of Mentalese is its independence from the 'public language'. That is, the hypothesis separates a language we speak in from a language we think in. Another is compositionality[10], which, paraphrasing Wilhelm von Humboldt's statement about language, allows to express "infinitely many things with finitely many means". Lastly, Mentalese can arguably be seen as a computational representation of thoughts. Such a trait of Mentalese is particularly appealing in building intelligent machines, especially if one agrees that thoughts are closely related to meaning.

Semantic parser Semantic parser maps a textual input into its formal representation – its meaning. Predicate logic, lambda calculi, Prolog or SQL formulas can serve as the final symbolic representation of the otherwise textual utterance. Historically, we can roughly distinguish three generations of semantic parsers. The first generation of semantic parsers is rule-based and has been used in Natural Language Understanding systems such as SHRDLU [Winograd 1971]. In the second generation, semantic parsers are induced (or learnt) from symbolic forms [Zettlemoyer and Collins 2007]. Learning-based approaches have greatly improved generality of the semantic parsers, and have also partially liberated designers of thereof from modeling errors. However, such semantic parsers are still trained from textual utterances associated with the symbolic counterparts. This makes the annotation effort quite big, as the annotators need to be familiar with logic. Obviously, relying on costly annotations limits the applicability of the parser to different domains This issue has been handled by the most recent generation of semantic parsers that are learnt from textual input and output pairs for the question answering task [Liang et al. 2013]. Semantic parsers, arguably, follow the Mentalese school of philosophy, with a well structured, compositional symbolic representation of the meaning.

2.3.2 Sub-symbolic Representation of the Meaning

In the following, we briefly describe alternative approaches to represent the meaning. They form a foundation to our neural-based method presented in Chapter 10 and Chapter 11. We first illustrate a linguistic hypothesis of how the meaning of words is formed. Next, we briefly describe a computational approach to represent the meaning of words. This tool is consistent with the aforementioned hypothesis. Finally, we describe another sub-symbolic approach to represent the meaning.

Distributional hypothesis Distributional hypothesis defines the meaning of a word "by a company it keeps"[11]. Intuitively, words that appear in similar contexts should have similar meanings. As opposite to Mentalese, such the hypothesis grounds the meaning of words in the 'public language'.

Words embedding The skip-gram variant of Word2vec [Mikolov et al. 2013], where the objective is to predict a context of a surrounding the input word, is a computational

[10] The meaning of the whole depends on a grammar and the meaning of its parts. The following is a more precise example of the compositionality: if $N \rightarrow ARB$ then $[\![N]\!] = [\![R]\!]\,([\![A]\!], [\![B]\!])$.

[11] John Rupert Firth

realization of the distributional hypothesis. By predicting the context of the given word, the architecture arguably learns a good representation of the given word that is implicitly characterized by its accompanying words – the gist of the distributional hypothesis. In this thesis, we however often generalize the notion of words embedding and use it in the context of any linear transformation of the word into a dense vector representation, or the output of such a transformation (a dense word vector itself).

Recurrent Neural Networks To represent larger linguistic entities such as phrases or sentences, vector representations (embeddings) of words have to be combined in some way. Simple summing of words representations leads to a Bag-of-words representations of the sentence. Such a representation however destroys an order that exists in the sentence. In order to maintain the order, Recurrent Neural Networks encode embedded words into a hidden representation $\boldsymbol{h}$ that is next passed to the subsequent step together with the upcoming word embedding realizing the equation: $\boldsymbol{h}_{t+1} = f(\boldsymbol{x}_t, \boldsymbol{h}_t)$, where a function f models temporal dynamics.

2.3.3 Concluding Remarks

Semantic parsers and Recurrent Neural Networks approach the problem of representing the meaning from fundamentally different perspectives. The former relies on hand-designed grammatical rules and a hand-defined set of predicates. On the contrary, the latter is a data-driven approach that does not require strong modeling assumptions. As we will see later in the thesis, both approaches have pros and cons. Sub-symbolic semantics seem to be better at capturing the combined visual and linguistic world, but we have not observed that they cooperate well with logical operators such as negation or counting.

2.4 Holistic Tasks

Not only should the hypothetical holistic machine perceive the world through books, but should also perceive visual aspects of the world. Therefore, the holistic machine could understand the world through a textual and visual inspection. Even further, such the holistic machine would be engaged in numerous discussions about the world, integrating both visual and textual knowledge to reason about different aspects of it.

In the remainder of this section, we outline a few tasks that combine vision with language, arguably a necessary ingredient in building holistic machines. In particular, we describe the 'question answering about images' task, which is an important subgoal of a broader Visual Turing Test. This task is further detailed in the majority of this thesis (Chapters 8, 9, 10, 11, Appendix B and Appendix C). Next, we outline a few challenges that holistic tasks impose on machines.

2.4.1 Combining Vision with Language

Due to the recent progress in Computer Vision and Natural Language Understanding, a growing body of the recent work focuses on a multimodal scenario by tackling the following

tasks: Zero-Shot Learning, Image-to-Text Retrieval, and Image Description Generation. All such tasks precede our work on the Visual Turing Test, which is illustrated as the last one and compared with the other three aforementioned tasks.

Zero-Shot Learning Although image classification has shown a tremendous progress in recent years [Russakovsky et al. 2014], such approaches need quite a lot of training data covering all the classes of our interest. In contrast, Zero-Shot Learning presents a scenario where no training examples of a particular class are shown to a learnable architecture. However, a knowledge about the category can still be transferred through its description. For instance, with a detailed description of an unknown bird such as the bird's attributes, the machine can correctly recognize the bird even if it 'sees' it for the first time [Lampert et al. 2009; Rohrbach et al. 2011; Akata et al. 2016].

Text-to-Image Retrieval Image-to-Image Retrieval searches for an image in a database of images that is the best described by a textual input [Lan et al. 2012; Karpathy et al. 2014; Malinowski and Fritz 2014c]. Similarly to Zero-Shot Learning, the problem asks for a good multimodal embedding that maps visual and textual data into a common space, in which similarities between both modalities become meaningful.

Image Description Generation Through the research on Visual Recognition and Natural Language Understanding, methods that describe the content of a given image have been developed [Vinyals et al. 2014; Xu et al. 2015; Donahue et al. 2015]. Most of the current architectures can be seen as encoder-decoder methods, where a visual information is first encoded into a representation from which a decoder generates a description. In contrast to retrieval tasks, the ultimate goal is to create novel descriptions of images rather than to find a one from a set of pre-defined textual descriptions that matches well with the image.

Question Answering about Images The recent successes in Visual Recognition, Natural Language Understanding, as well as advances in multimodal embedding have led us to develop a holistic task that challenges intelligent machines with a holistic scene comprehension by asking questions about the content of images that is a part of a broader Visual Turing Challenge. A good performance on such a question answering task is, arguably, a necessity in developing holistic machines. It also shares many properties with all the aforementioned multimodal tasks. Most of all, the task requires finding a multimodal mapping between language and the vision. Similarly to Zero-Shot Learning, not all questions or answers are encountered during training. The Visual Turing Test can also be seen as a generalization of the Text-to-Image Retrieval task, where the retrieved images have to satisfy conditions imposed by a question. Finally, the Visual Turing Test is a more focused variant of the Image Description Generation, where generated answers are conditioned on both image and textual inputs, and hence they have to directly fulfill the conditions imposed by a questioner.

2.4.2 Challenges

As we develop machines that solve harder and more holistic problems, new challenges occur. Holistic problems are necessarily multimodal, where reasoning should be done not only over visual and linguistic domains, but likely even more senses are needed. This poses a question of a suitable representation that, arguably, has to be learnt from data. The architectures need to reason about more and more concepts with fuzzy semantical boundaries – this contrasts with traditional classification tasks where mutually exclusive categories are often assumed. Finally, holistic machines need to work with different sources of ambiguities as these are inherent to 'human world'. Chapter 8 enumerates in greater detail challenges that we find important for the Visual Turing Test.

2.4.3 Concluding Remarks

Over the last few years we have seen a tremendous progress in image classification, and a major shift towards data-driven, jointly trained, deep recognition architectures. Similar approaches can also be successfully used to handle other tasks such as object detection, image segmentation, or natural language modeling. Such success stories provide a strong evidence on the generality of Deep Learning. However, they have also opened a question of the role of Deep Learning in handling more holistic tasks like the Visual Turing Test. In such tasks a joint understanding of a scene, language, human intentions, common-sense knowledge, and logical reasoning all play an important role. Chapter 10 and Chapter 11 investigate this role, and compare with a logical-based approach presented in Chapter 9.

Chapter 3

Background: Visual Recognition

Contents

Human ability to efficiently process visual information such as shapes, or colors in the surrounding environment together with associating semantic attributes to observed objects play an important role in acquiring knowledge in every-day life. Therefore, it is unsurprising that we expect a similar behavior from intelligent machines, and therefore mimicking (at least in a behavioral sense) of the human visual system has a long-standing tradition in the Computer Vision community. Moreover, every bit of progress in the development of the visual recognition, presumably, also translates to better performance in other, more complex tasks requiring processing a visual input such as detection, image captioning, or question answering about images [Ren et al. 2015b; Venugopalan et al. 2015a; Malinowski et al. 2016]. This is also explored in Chapter 11 for the latter. Over the most recent history of Computer Vision, two approaches to recognition have been particularly important: Spatial Pyramid Matching (SPM) and Convolutional Neural Networks (CNN). Both methods play an important role in later chapters of the thesis, and are introduced in greater detail in this chapter.

3.1 Introduction

Origins of visual recognition can be found in the early 1960s [Roberts 1963]. Pioneering approaches have attempted to reduce complex visual world to geometrical primitives [Roberts 1963; Agin and Binford 1976; Biederman 1987], emphasizing that geometrical shapes of objects convey the notion of objects themselves. Together with the progress on the color constancy problem, the problem of perceiving identical colors under different light conditions, later methods have stressed the importance of color in the identification of objects [Swain and Ballard 1991]. This can be seen as the root of the appearance-based methods, in which color or pose becomes an important cue in the machine recognition [Swain and Ballard 1991; Murase and Nayar 1995]. Roughly at the same time, learning-based architectures have started to dominate the field [Poggio and Edelman 1990]. Most recent recognition methods,

such as the Spatial Pyramid Matching and Convolutional Neural Networks architectures, assume a hierarchical organization of an effective recognition system.

Before delving into details of different recognition architectures, let us first formalize the notion of visual recognition. For the purpose of this thesis, we are mostly interested in visual recognition restricted to a classification problem. Hence, recognizing an object in the given image means assigning a class (most often a noun) from a set of all possible classes to the image. Formally, the recognition is a function ϕ that maps images into classes (categories), that is $\phi(\boldsymbol{x}) = c$ where $\boldsymbol{x}$ is an image, and $c \in \mathcal{C}$ is a class that belongs to a set of all classes $\mathcal{C}$. For years, the most successful approaches to machine recognition assume that a function ϕ comes from some function space $\mathcal{F}$ but otherwise is unknown, and therefore must be learnt from a finite training set $\mathcal{T}_{\text{train}} := \{(\boldsymbol{x}_j, c_j)\}_j$ consisting of samples representing a problem of our interest. Training of recognition architectures is typically conducted by finding a function that minimizes an expected risk on the training set $\mathcal{T}_{\text{train}}$, that is $\phi^* := \arg\max_{\phi \in \mathcal{F}} \mathbf{E}_{\mathcal{T}_{\text{train}}} [\ell(\phi(\boldsymbol{x}), c)]$. To test how methods generalize to unseen examples, accuracies are reported on the separated test set $\mathcal{T}_{\text{test}}$ [1], in which images are classified according to a class assignment procedure, often $c_{\boldsymbol{x}} := \arg\max_{c \in \mathcal{C}} \ell(\phi^*(\boldsymbol{x}), c)$ for all $(\boldsymbol{x}, c) \in \mathcal{T}_{\text{test}}$. Such a protocol follows a standard paradigm of machine learning.

Recently, due to the increase of computational resources as well as availability of large volume training datasets $\mathcal{T}_{\text{train}}$ (see also Section 2.2), we have seen a departure of the research community from hand-designed architectures such as Spatial Pyramid Matching (Section 3.2) towards jointly trained, end-to-end convolutional architectures (Section 3.3). The latter approaches to recognition have dominated the field achieving near human-quality performance on a particularly popular, large volume recognition dataset – ImageNet.

In the remaining of this chapter, two popular approaches to visual recognition are juxtaposed. Both methods also play an important role in later chapters. In Chapter 7, we extend the Spatial Pyramid Matching architecture (Section 3.2) to learn a spatial division together with the classification objective by a joint optimization strategy. In Chapter 10, we build our neural-based visual question answering architectures based on Convolutional Neural Networks (Section 3.3) that are currently the leading visual recognition architectures.

3.2 Spatial Pyramid Matching (SPM)

Spatial Pyramid Matching (shortly SPM) is, based on the locally orderless images idea [Koenderink and Van Doorn 1999], a framework of building a more abstract global representation of an image from local patches. Although building such a representation by computing a histogram (a pooling step) of local representations has already been used in orderless Bag-of-Visual-Words methods, Spatial Pyramid Matching preserves some spatial information by spatially dividing the image into large subregions [Lazebnik et al. 2006; Yang et al. 2009; Coates and Ng 2011]. At the very first step, patches from all images in the training set are collected and categorized into K categories in an unsupervised way – for instance, with K-means clustering or sparse coding algorithms – to form a dictionary of

[1] We assume the following relation between both sets: $\mathcal{T}_{\text{train}} \cap \mathcal{T}_{\text{test}} = \emptyset$. Note, however, that the set of all classes $\mathcal{C}$ remains the same. This may not hold in other tasks such as Zero-shot Learning.

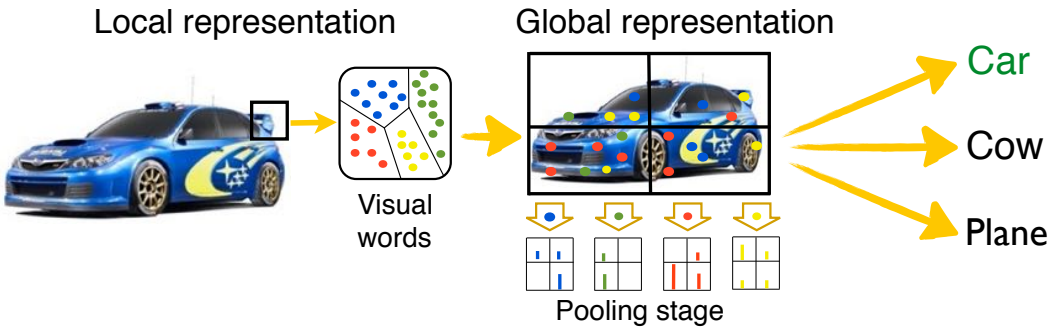


Figure 3.1: Patches are encoded into visual words (the local representation) that are next spatially gathered to form a subregion representation of the image. All the subregion representations are next concatenated, and serve as a global representation of the whole image. Such a representation becomes a feature vector used by a classifier. The latter chooses a correct class from the set of all classes (here, it chooses from 'Car', 'Cow', 'Plane').

visual words $\boldsymbol{V}$. Sometimes, before the clustering step, a local descriptor such as SIFT [Lowe 2004] is applied to every patch. Next, as Figure 3.1 sketches, local patches collected from each image are encoded into visual words according to the dictionary $\boldsymbol{V}$. The pooling stage concatenates pooled (sum or max pooling is often used) visual words from all L spatially divided subregions (Figure 3.1 show a 2-by-2 spatial division of the image) to form a global representation of the image – a $K * L$ dimensional feature vector – that is fed into a classifier. The pooling stage can also form a pyramid of different global representations with an increasing number of spatial subregions. For instance, Lazebnik et al. [2006] consider 3 levels of pyramid with the following divisions: 1-by-1, 2-by-2, and 4-by-4. Forming such pyramids is motivated by the observation that different levels increase invariances such as translation invariance by disregarding locally spatial information. For example, although 1-by-1 division, conceptually equivalent to a Bag-of-Visual-Words method, is the most destructive, this level is also fully invariant to translations. Another possibility to preserve the information and free the method from arbitrary divisions is to use a weighted pooling operator and learn the division discriminatively together with a classifier [Malinowski and Fritz 2013b,a].

Conceptually, SPM can also be seen as a dimensionality reduction technique that maps a high dimensional input image into $K * L$ dimensional feature vector. SPM can also be used as a 'trick' to fix dimensionality of the global feature vector making the method independent of the number of sampled patches or the size of the image. Finally, some low level vision descriptors such as HOG or SIFT [Lowe 2004; Dalal and Triggs 2005] can also be viewed as particular instances of SPM. Under such an interpretation, patches are divided into bins (division into subregions) and histograms are computed over gradients (pooling).

This framework differs from the next generation of recognition architectures (Section 3.3), namely Convolutional Neural Networks, mainly by lack of joint training. That is, every stage in the architecture – namely a sampling scheme used to collect patches, type of local descriptors applied to the patches, a dictionary learning and encoding methods, a pooling operator[2], a spatial division, number of levels in the pyramid, an employed classifier – is

[2]Two operators are particularly popular: computing maximum or summation over the visual words within

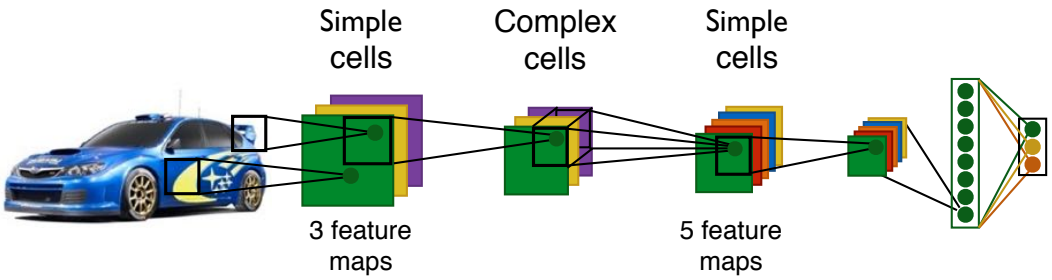


Figure 3.2: Receptive fields of simple cells are tiled to cover the whole image (convolutions). A linear mapping of the patch ('cropped' input according to the receptive field), followed by a nonlinearity becomes the response of the simple cell (two simple cells connected with the image are singled out in the figure for the visualization purpose). All the cells from the same feature map (here: green, yellow, and violet) share the weights. Different neurons from the same feature map have access to different parts of the image. The last feature maps are densely connected to a dense layer, followed by a classifier (colors in the last layer correspond to three classes: 'Car', 'Cow', and 'Plane').

designed and learnt/computed separately. It is worth mentioning that there are variants of the architecture that replace an arbitrary spatial division step with a learnable scheme, e.g. the one presented in Chapter 6.

3.3 Convolutional Neural Networks (CNNs)

CNN Convolutional Neural Networks (shortly CNNs) have recently replaced more traditional, hand-designed recognition architectures. Although they share many basic ideas with the Spatial Pyramid Matching framework, most prominently the idea of building successively more global representations from the local ones (compare Figure 3.1 to Figure 3.2), all layers in CNNs, as opposed to SPM, are jointly trained by back-propagation [LeCun et al. 1998a] and each module (e.g. a layer of simple cells followed by a layer of complex cells is a single module in Figure 3.2) is, conceptually, the same. Nowadays, owing to better hardware, advances in initialization of Deep Convolutional Neural Networks consisting of many consecutive layers, and progress in stochastic training, CNNs can be trained end-to-end, directly from pixels to classes on large volume datasets.

CNNs are composed of two kind of cells that are called, drawing inspirations from neuroscience, simple and complex cells [Hubel and Wiesel 1962; Fukushima 1980; LeCun et al. 1998b; Serre et al. 2007]. Simple cells respond to some patterns like edges within their receptive fields[3], whereas complex cells, with larger receptive fields, bring some sort of invariances to where the pattern is observed and reduce the dimensionality of previous stage. The former is computationally realized as responses to linear filters applied to the input within the corresponding receptive field, followed by some nonlinearity, while the latter is implemented as a pooling operator, already seen in SPM that is computed over

a subregion.

[3]Receptive field is a subregion in the input that the cell is sensitive to.

smaller regions and hence less destructive. Since the pattern can be observed anywhere in the visual input, the receptive fields are tiled to cover the whole input. Such responses of simple cells over the all tiled receptive fields are mathematically expressed by convolutions. More formally, the simple cell response is $z_j = \sigma(\boldsymbol{w}^T \boldsymbol{x}_j + b)$, and the complex cell response is $c = \rho(z_1, x_2, ..., z_k)$. Here, $\boldsymbol{x}_j$ is the j-th patch sampled from the input, ρ is a pooling operator, and σ is a nonlinearity; often $\rho = \max$, and $\sigma(x) = \frac{1}{1+e^{-x}}$. Note that the size of $\boldsymbol{x}_j$ and k are determined by simple and complex's receptive fields respectively. Tiling of the receptive fields can be formalized using convolutions. All the neural cells with translated receptive fields form a feature maps. All the weights connected to the same feature map are shared, and hence they respond to the same, but spatially translated, pattern. The weights are not shared across different feature maps, and so typically different feature maps respond to different patterns. So that $z = \sigma(\boldsymbol{w}_k^T \boldsymbol{x}_j + b_k)$ where $\boldsymbol{w}_k$ and b_k are parameters from the k-th feature map. As Figure 3.2 suggests, the dot product operations $\boldsymbol{w}^T \boldsymbol{x}_j$ can also be generalized to tensor multiplications in order to accommodate for computing a response over many feature maps. Such a module can be repeated, where the input image $\boldsymbol{x}$ is replaced by a complex cell response map $\boldsymbol{x}^l$ from the l-th layer. With this notation, we have $z_j^{kl} = \sigma(\boldsymbol{w}_k^{lT} \boldsymbol{x}_j^{l-1} + b_k^l)$ with $\boldsymbol{x}_j^0$ representing the input image, and $\boldsymbol{x}^l = [z_j^{kl}]_{jk}$ (for the sake of simplicity, we ignore complex cell responses in the equation). Finally, the last convolutional module is flattened, and densely connected with one or two dense layers, just before such responses are given to a classifier. The whole architecture is trained end-to-end with back-propagation [LeCun et al. 1998a].

Interestingly, CNNs can also be seen as a features extractor technique that can be used by methods working on other tasks requiring processing of the visual input (in Chapter 10 we will see such an application of CNN). That is, once a network is trained, the responses from some layer (often the last dense or convolutional layer is used) are abstract representation of the input image and therefore are extracted to serve as a feature vector in other methods.

Finally, CNNs can, presumably, effectively exploit large-volume datasets, and their performance does not saturate as quickly as the performance of earlier methods. That is, they exhibit scalability both in computability, and learnability (Section 2.1). As a result, for about 3 years CNNs are driving advances in recognition, however, each year sees new variants of CNNs that furnish the plain version described in this section with new components.

AlexNet Until 2012, the most successful approaches to recognition relied on hand-designed features. Combinations of different classifiers with SIFT, HOG and Fisher Vectors topped the ladder of the ImageNet challenge [Russakovsky et al. 2014]. This, however, has been changed due to an architecture introduced by Krizhevsky et al. [2012], which outperformed other approaches by about 10 percent points achieving 15.31% top-5 error.

The presented model, called AlexNet and shown in Figure 3.3, is a deep CNN approach to recognition that is trained end-to-end, from RGB pixel values to classes. It has five convolutional layers, most of which are followed by max-pooling layers, three fully-connected (or dense) layers and a final 1000-way softmax (Figure 3.3). In total it has about 650,000 neurons with 60 million parameters. To make training of such a large network feasible, the network uses novel non-saturating Rectified Linear Units (ReLU) [Nair and Hinton

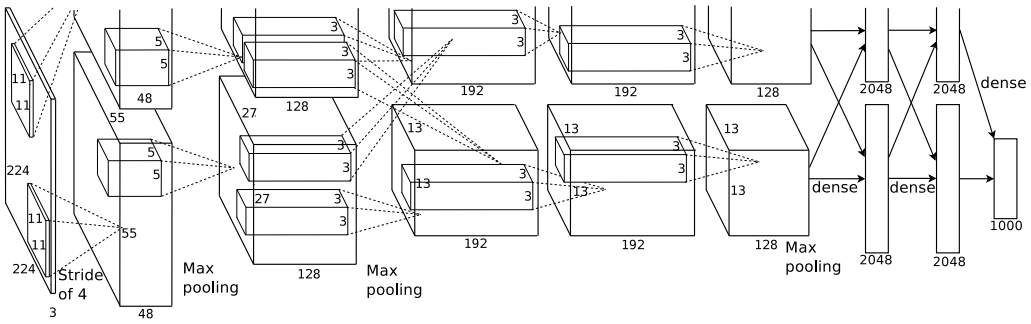


Figure 3.3: AlexNet CNN. Note that the delineation of the network in the figure is due to its parallelization to take advantage of two GPUs. The figure is taken from Krizhevsky et al. [2012].

2010], that is $\sigma(x) = \max(0, x)$. The objective function that the network maximizes is a multinomial logistic regression.

For the matter of being more detailed about the AlexNet architecture shown in Figure 3.3, the network's input is an RGB 'crop' (with the size 224-by-224-by-3) of the original image that is followed by a convolutional layer with 96 feature maps and a convolutional kernel of the size 11-by-11-by-3 and stride of 4. A max-pooling layer reduces the size of the convolutional layer so that each outputted feature map has the size 55-by-55. The second convolutional layer convolves the previous layer with a convolutional kernel of the size 5-by-5-by-48, and is followed by a max-pooling layer. This results in the 256 feature maps with the size 27-by-27 each. The following two convolutional layers have 384 feature maps each, and convolve their inputs with convolutional kernels of sizes 3-by-3-by-256 and 3-by-3-by-192 respectively. The last convolutional layer has 256 feature maps and a convolutional kernel of the size 3-by-3-by-192. It is also again followed by a max-pooling layer. The last two densely connected layers have 4096 neurons each. The last densely connected layer is finally connected to the 1000-way softmax.

To improve generalization, Krizhevsky et al. [2012] suggest using Local Response Normalization, which implements a form of a 'lateral inhibition' between neurons from different feature maps, together with a novel regularization layer called Dropout [Hinton et al. 2012]. At every training iteration, the Dropout layer sets independently and with probability p a response of each neuron to zero, effectively removing the neuron from the network. This is a form of the stacking technique applied to neural models where at each training iteration a model is sampled from exponentially many possible models that share the same parameters. Note that higher p puts more pressure on single neurons to have significant responses towards input patterns and to 'play well' with other unknown neurons reducing its co-adaptation. This results in an ensemble of simpler models that each one is less prone to overfitting. At test time, we compute an approximation of the expected response of the whole network by multiplying each neuron by a constant value p. Krizhevsky et al. [2012] use $p = 0.5$.

Since using more training data often leads to better generalization, Krizhevsky et al. [2012] augments training data with three class-preserving transformations: image translations, horizontal reflections, and pixel intensities alternation. The first two transformations are

implemented as random 224-by-224 'crops' and the horizontal reflections of the input image. At test time, the softmax predictions are averaged over five crops (at the center and four corners) extracted from the image together with their horizontal reflections. The pixel intensity alternation scheme captures the observation that object's classes are invariant under changes in the intensity and the color of the illumination [Krizhevsky et al. 2012].

All the aforementioned ingredients have greatly improved generalization on the ImageNet classification challenge, where the model has outperformed the competitors by a large margin, and hence have changed the way how we design and train recognition architectures.

3.4 Recent Recognition Architectures

After the AlexNet winning entry in 2012, it becomes clearer that depth is essential in developing modern CNNs. The most recent variants of CNNs extend AlexNet with novel network's designs that allow for training deeper models.

GoogLeNet GoogLeNet [Szegedy et al. 2015] builds an Inception module (Figure 3.4) by performing convolutions at different scales together with dimensionality reduction through 1-by-1 convolutions. The 1-by-1 convolutions allow to efficiently train deeper and wider architectures. To combat with the gradient vanishing problem, in addition to ReLUs, extra classifiers are coupled with some Inception modules that provide a supplementary signal of supervision to lower layers (Figure 3.5).

VGG-net VGG [Simonyan and Zisserman 2015] uses narrow and repeated convolutional kernels of the size 3-by-3, and stride 1 pixel to build a deep CNN. Both GoogLeNet and VGG have achieved remarkable performance on the ImageNet 2014 classification challenge: about 6.7% and 7.3% top-5 error respectively. With depth[4] 22 and 19 respectively, both architectures are also significantly deeper than AlexNet (with depth only 8).

Residual Net The next two networks achieve top-5 error about 3.5% on the ImageNet 2015 classification challenge. He et al. [2015] have increased depth of the network to 152 layers. To successfully train such a deep network, the authors have introduced Residual Net with shortcut connections (Figure 3.6), where it is always easy to learn a perturbed identity transformation. The architecture uses 1-by-1 convolutions that perform dimensionality reduction. Interestingly, the design of Residual Net departs from the traditional design that the previously mentioned models exhibit. The network does not use pooling, there is no dropout, nor a hidden fully connected layer. The ensemble of Residual Nets achieves 3.08% top-5 error. Table 3.1 aggregates results of different networks on the ImageNet datasets.

Concluding Remarks All the recent developments of CNNs show a clear trend that deeper networks work better. This is summarized in Table 3.1. However, training deeper networks comes with the cost. First, the gradient of very deep networks vanishes causing difficulties in training their lower layers. Second, deeper networks are computationally more

[4]Here, depth is defined as the number of consecutive layers with parameters.

Name	Depth	Top-5 Error
Pre-CNN	-	26.17%
AlexNet	8	15.31%
VGG-Net	19	7.3%
GoogleNet	22	6.7%
ResNet	152	3.5%

Table 3.1: Performance of different, popular variants of CNNs on the ImageNet dataset. Pre-CNN refers to the ISI team with the best method from the ImageNet Challenge 2012 that does not use CNNs.

expensive to run. The very or ultra deep networks handle the aforementioned issues with a clever design. For example, narrow convolutional filters, loss linked to lower layers, lack of hidden fully connected layers, and shortcut connections are just a few possible remedies. The three aforementioned networks are collectively illustrated in Figure 3.7.

3.5 Conclusion

This chapter summarizes two popular approaches to image classification that share important properties: Spatial Pyramid Matching, and Convolutional Neural Networks. Both have left a mark in the history of Computer Vision, both stress an importance of hierarchies of representations to build a global image representation, and both use pooling over regions. However, at the same time, there are important differences between both frameworks. Convolutional Neural Networks pool over much smaller receptive fields, and are jointly trained, end-to-end architectures. Whereas Spatial Pyramid Matching pools over large spatial regions, where each stage (layer) is either hand-designed or separately trained. Chapter 6 makes the gap between both methods smaller by considering a variant of SPM with two layers of the architecture that are jointly trained. Nonetheless, CNNs are currently driving advances in the visual recognition, and hence are extensively used in the thesis.

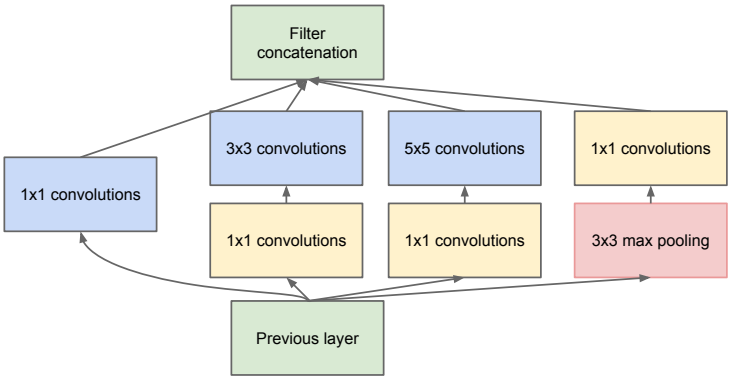


Figure 3.4: Inception module uses convolutions at different scales together. The figure is taken from Szegedy et al. [2015].

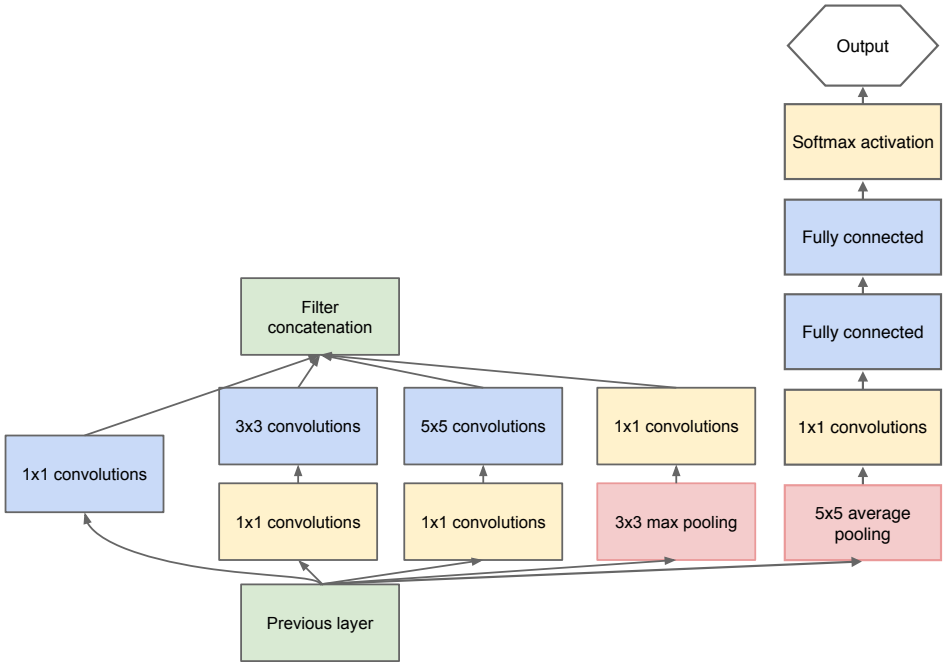


Figure 3.5: Inception module with a classifier that provides an additional signal of supervision. The figure is based on Szegedy et al. [2015].

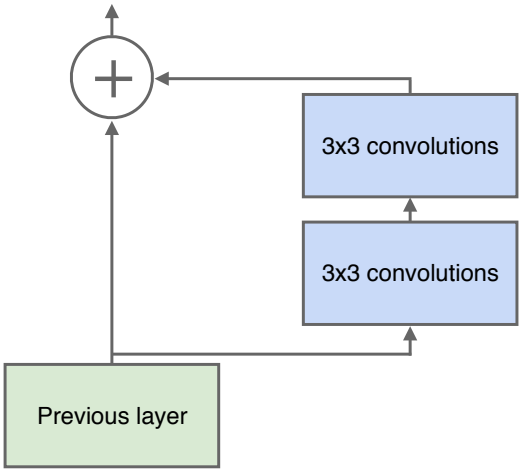


Figure 3.6: Residual Net with the shortcut connections.

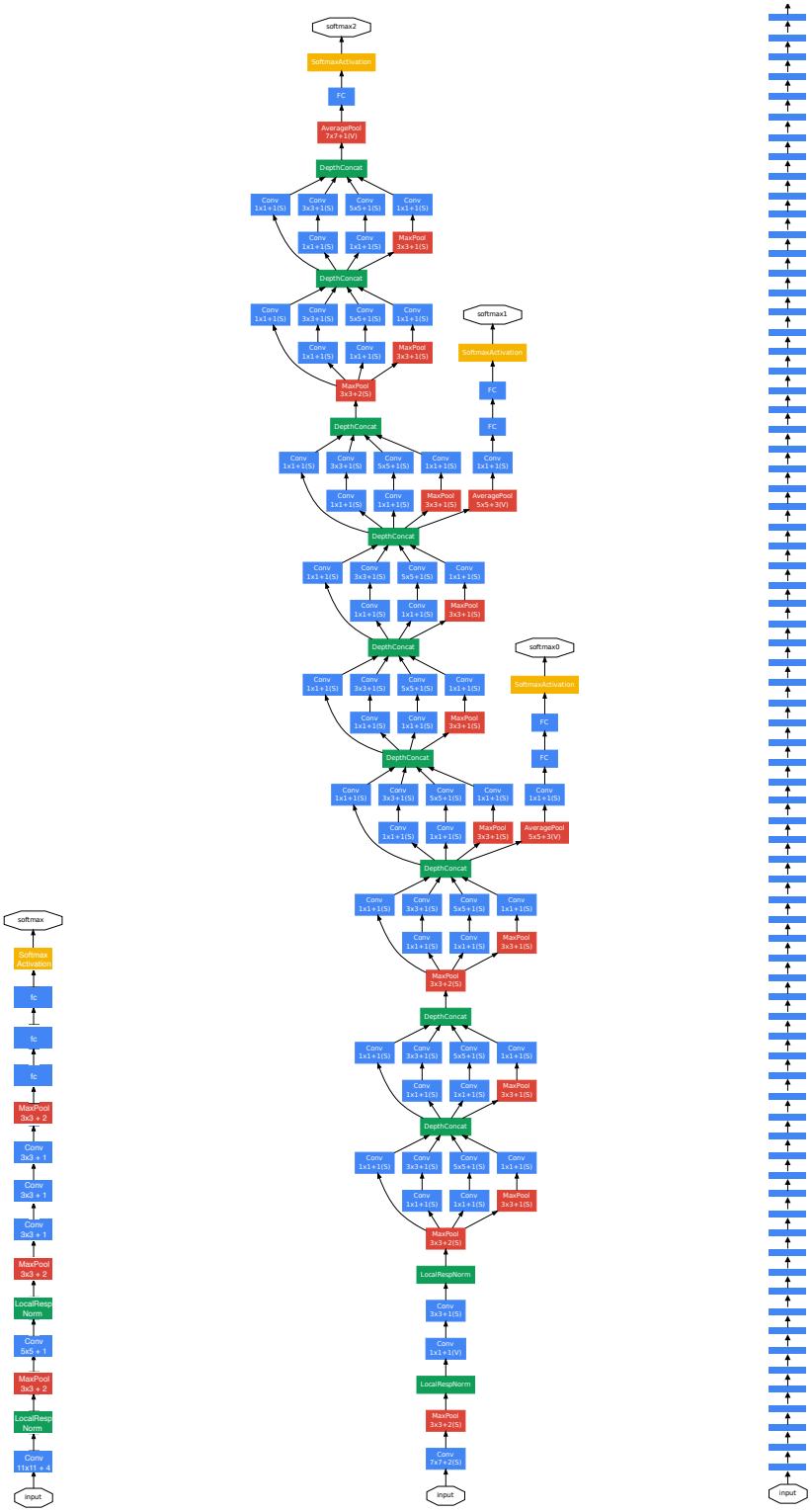


Figure 3.7: Depiction of deep, very deep, and ultra deep networks. From the left: AlexNet (depth 8), GoogLeNet (depth 22), and Residual Net (depth 152 - due to the space limit, the depiction is incomplete). The figure is based on Szegedy et al. [2015] and He et al. [2015].

Chapter 4

Background: Natural Language Understanding

Contents

Human ability to efficiently process linguistic information by associating a meaning to words, phrases, sentences is the next fundamental modality that plays an important role in acquiring knowledge in every-day life. Languages have evolved together with us, and remain an important communication channel. Therefore, we would also like to communicate with intelligent machines through the language. While communicating with others, we do not say everything but rather we rely on a common (unspoken) ground with the receiver. Similarly, according to our preferences, we may interpret words, phrases, or sentences in different ways. This is a source of many ambiguities that we can encounter in language, which machines have to deal with too. In this chapter, we discuss two seemingly different approaches to handle language understanding: logical-based and neural-based. Both are used as building blocks of holistic machines that answer questions about images. Both are also discussed in Chapter 9 and Chapter 10 in the context of the 'question answering about images' task.

4.1 Introduction

Natural Language Understanding dates, arguably, back to the sixties of the twentieth century. For example, Bobrow [1964] have designed a program that solves high school algebra homework. Another famous project is SHRDLU [Winograd 1971], where a block world consisting of some geometrical primitives is provided with the goal of rearranging the blocks in a certain way by controlling a robot's hand through natural language instructions (Figure 4.1 shows the 'block world' that SHRDLU operates on). However, projects from that period of time had limited scope of applications as a result of using hand-designed translation rules. Much later, approaches to natural language understanding can roughly be categorized into logic-based approaches that use semantic parsers to translate a textual input into some formal representation, or neural-bases approaches that transform the textual input into

some vector representation. Both approaches to represent the meaning are also investigated in Visual Turing Test, and yield two seemingly different methods. In this chapter, we will visualize both approaches to represent the meaning of a sentence in greater detail. The first part concerns semantic parsing. First, we introduce the general idea that stands behind the semantic parsing, next we briefly provide a brief historical context concerning the semantic parsing, show how to train a simplified semantic parser (based on the work of Liang and Potts [2015]), and finish with a brief introduction to Dependency-Based Compositional Semantics introduced by Liang et al. [2013] that is used in our first approach to answer questions about real-world images, presented in Chapter 9. The second part concerns with Recurrent Neural Networks, which have recently gained popularity in processing textual input. We use the latter approach to language modeling in our second approach to answer question about real-world images, presented in Chapter 10 and Chapter 11. Indirectly, a similar basic idea stands behind approaches to retrieve images based on textual queries that are described in Chapter 7 and Appendix A. A curious reader may also read over our tutorial presented in Appendix C.

4.2 Semantic Parsing

Overall picture Semantic parsing is footed on the desire to have an automatic tool that fully specifies the meaning of a sentence by representing it in a formal language. Such a formal representation must be sufficient to complete a task. For instance, it must be compatible with formal ways to achieve a final destination in a robot planning scenario [Tellex et al. 2011], or to withdraw an answer from a database in the 'question answering with a Knowledge Base' task [Liang et al. 2013]. Since in this thesis we use a semantic parser to represent questions in the 'question answering about real-world images' task (Chapter 9), in this section, we focus on a similar 'textual question answering with a Knowledge Base' task with a semantic parser of Liang et al. [2013] as an exemplar one. All the further references to the semantic parser in this chapter refer to the aforementioned case.

The task that we are interested in this chapter assumes a Knowledge Base that is a formal representation of a domain knowledge, as well as textual question-answer pairs that serve as a signal of a supervision. The Knowledge Base, which we also call a World, is often implemented as a database storing certain facts about the real-world. For instance, the Knowledge Base can store geographical facts or jobs descriptions [Liang et al. 2013].

Semantic parsing for question answering Let $[\![s]\!]_{\mathcal{W}}$ denote the meaning of a sentence s with respect to the Knowledge Base $\mathcal{W}$, and let $\ell(s)$ be the formal representation of the sentence. Then, with a slight abuse of the notation, $[\![\ell(s)]\!]_{\mathcal{W}}$ be the interpretation of the formula $\ell(s)$ with respect to $\mathcal{W}$. Depending on the exact design decisions about the formal language and the Knowledge Base, $\ell(s)$ can be a SQL formula produced by the semantic parser, and $[\![\ell(s)]\!]_{\mathcal{W}}$ would be the result of executing the SQL formula on the database $\mathcal{W}$. It is also worth noting that in this scenario, the logical formula $\ell(s)$ is already unambiguous, and hence, the interpretation $[\![\ell(s)]\!]$ is a deterministic mechanism. This, however, contrasts with the 'question answering about real-world images' task, in which the input is inherently

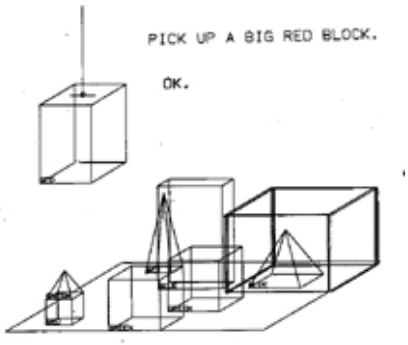


Figure 4.1: The 'Block World' of SHRDLU. The figure comes from http://hci.stanford.edu/winograd/shrdlu/.

uncertain due to various (human or machine) interpretations of the scene. Therefore, in Chapter 9 we extend the interpretation mechanism to work with probabilistic Knowledge Bases that represent uncertainty in the visual input.

Historical outline So far we have taken semantic parsing for granted by assuming that the parser somehow transforms strings into logical formulas. But how is it done? Historically, first semantic parsers use hand-designed rules to do the transformation. In the early seventies, Winograd [1972] developed SHRDLU, an early conversation machine connected to the 'block world' (see Figure 4.1). Although the SHRDLU's performance was impressive within the toy-world domain, extending the architecture to broader domains has failed. Covering all nuisances of the natural language with hand-designed rules is, reportedly, very difficult [Liang 2014]. Due to the problems in scaling up rule-based parsers to other domains, the next generation uses machine learning approaches to induce a semantic parser based on training data, in which textual sentences are paired with the corresponding formal representations [Zelle and Mooney 1996; Zettlemoyer and Collins 2007; Kwiatkowski et al. 2010]. Although shifting to learning-based approaches has been an important step forward in achieving better scalability, the existence of logical forms representing utterances requires an expert knowledge and therefore increases the annotation effort, and limits the amount of possible training data points. To handle such issues, the research community has proposed learning-based approaches to natural language understanding that induce a semantic parser directly from question-answer (or utterance-denotation) pairs [Clarke et al. 2010; Liang et al. 2013]. Arguably, the most important property of the learning-based semantic parsers comes from the following observation. Since good derivations of the meaning are difficult to obtain, the parsers over-generate the space of good derivations by also allowing to have incorrect ones, and next rely on machine learning approaches to rank the derivations.

A simplified task of solving algebraic formulas To look closer at how semantic parsers are trained from utterance-denotation pairs, we examine a simplified task of learning to solve algebraic formulas from natural language descriptions. This part is inspired by the work of Liang and Potts [2015]. Let us consider a linguistic object $\langle s, t, l, i \rangle$ – with a sentence s, its syntactic derivation t, its logical representation l, and its inter-

Syntax	Logical Form
$N \rightarrow one$	1
$N \rightarrow one$	2
⋮	⋮
$N \rightarrow two$	1
$N \rightarrow two$	2
⋮	⋮
$R \rightarrow plus$	$+$
$R \rightarrow plus$	$-$
$R \rightarrow plus$	$\times$
$R \rightarrow minus$	$+$
$R \rightarrow minus$	$-$
$R \rightarrow minus$	$\times$
$R \rightarrow times$	$+$
$R \rightarrow times$	$-$
$R \rightarrow times$	$\times$
$N \rightarrow N_{\text{left}} R N_{\text{right}}$	$\lceil R \rceil(\lceil N_{\text{left}} \rceil, \lceil N_{\text{right}} \rceil)$

Table 4.1: Grammar for the algebraic formulas task. We follow a standard, mathematical interpretation of the logical forms. The table is a simplified version of the table shown in Liang and Potts [2015].

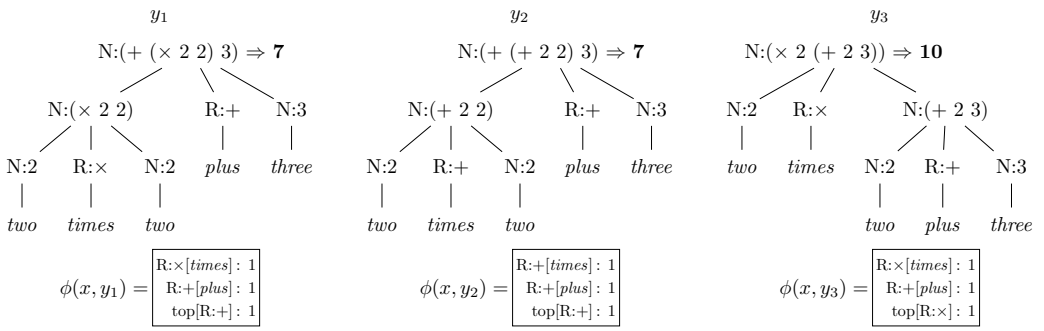


Figure 4.2: Results of the $Gen(x)$ procedure that generates logical formulas, illustrated here as parsed trees, from a sentence x. $\phi(x, y)$ is a feature function between the sentence x and its logical form y. Features include: number of times an operator appears in y ($R : \times[times]$ or $R : +[plus]$), and a binary indicator showing which operator is used as the root of the tree ($top[R : +]$ or $top[R : \times]$). The sentence x is 'two times two plus three'. The figure comes from Liang and Potts [2015].

pretation i – and training pairs of sentences paired with their denotations. For instance, $\mathcal{T} := \{(\text{two times three plus four},\ 10), (\text{two plus two}, 4), \ldots\}$. The simplified algebraic task is compelling as its syntax, logical representation, and interpretation are easy to derive (Table 4.1) or natural (the interpretation procedure is just a standard mathematical interpretation of the formulas that can easily be implemented in modern programming languages), so that we can focus more on the learning task. Table 4.1 already implies a few challenges that a semantic parser has to handle. First, the association between syntactic tokens and logical forms is unknown, for instance $N \to one$ can be associated with any single digit number. Second, correct logical forms are unknown during training as only their denotations are observed. They are, however, used in an objective function as latent variables. Third, different logical forms can have the same denotations. For example, both interpretations of the token *plus* as $+$ or $\cdot$ lead to the same denotation of the utterance 'two plus two'. A proper interpretation of the 'plus' token can only be learnt once more training data are available. Four, precedence of the operators is also unknown and has to be learnt from data. To train a semantic parser, Liang and Potts [2015] suggest the following learning framework with a latent Support Vector Machine objective [Schölkopf and Burges 1999]

$$\min_{\boldsymbol{w}} \sum_{(s,i)\in\mathcal{T}} \max_{l\in Gen(s)} \left[\boldsymbol{w}^T\phi(s,l) + \Delta(i, [\![l]\!])\right] - \max_{l\in Gen(s,i)} \boldsymbol{w}^T\phi(s,l) \tag{4.1}$$

where $\boldsymbol{w}^t\phi(s,l)$ is a score function of deriving the logical form l from the sentence s, $Gen(s)$ is a generator of all logical forms that can be derived from s, $Gen(s,i)$ is a generator of all logical forms that can be derived from s such that their interpretations are exactly i, and $\Delta(i, [\![l]\!]) = 0$ if interpretation of l is the same as i but otherwise 1. Equation 4.1 can be trained with Stochastic Gradient Descent [Bottou 2010] using the following upgrade rule at sample $(s,i) \in \mathcal{T}$

$$\boldsymbol{w} := \boldsymbol{w} - \alpha\left(\phi(s,\hat{l}) - \phi(s,l^*)\right)$$

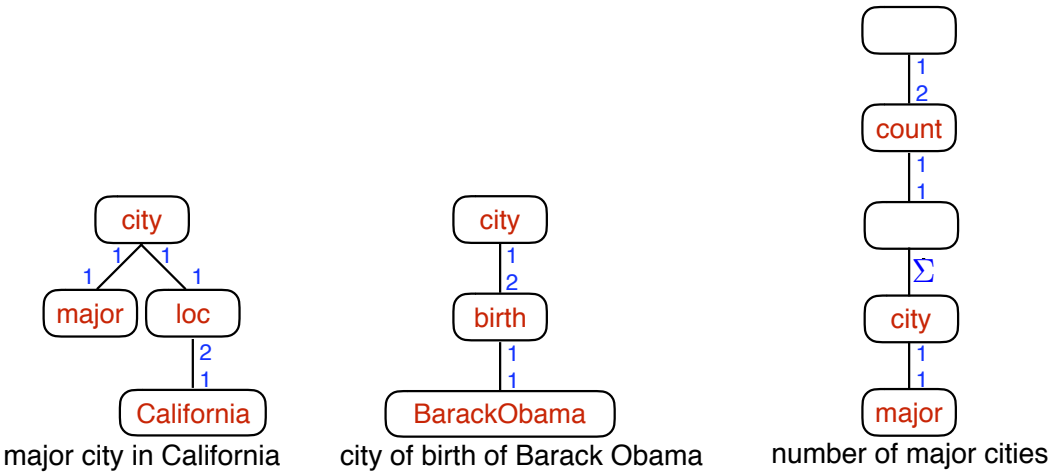


Figure 4.3: Examples of Dependency-Based Compositional Semantics trees. The figure is created based on the work of Liang et al. [2013].

where α is the learning rate chosen based on a validation set, $\hat{l} := \arg\max_{l \in Gen(s)} \boldsymbol{w}^T \phi(s,l) + \Delta(i, [\![l]\!])$, and $l^* := \arg\max_{l \in Gen(s,i)} \boldsymbol{w}^T \phi(s,l)$. The feature function $\phi(s,l)$ roughly captures different relationships between the textual input s and its logical representation l, such as the number of times a rule is used in the logical derivation, or which operator is used in the root. Figure 4.2 shows a few logical forms derived from the sentence "two plus two times three" together with used features. Note that, with 'stronger' grammatical rules, for instance by directly associating 'one' with 1, 'two' with 2, etc., we could significantly simplify the training effort. However, at the same time, we would loose the generality of the approach. In the domain of algebraic formulas this may work, but it is less likely to work well in much broader domains such as question answering about geographic facts [Liang et al. 2013] or Visual Turing Test [Malinowski and Fritz 2014a].

Dependency-based Compositional Semantics In Liang et al. [2013], the authors introduce Dependency-Based Compositional Semantics to efficiently encode logical forms (Figure 4.3 shows a few derivations), and use a probabilistic framework as the learning framework (shown in Figure 4.4). Despite of such changes, the core principles we have already presented remain the same. It is also worth mentioning that the *Gen* procedure in Equation 4.1 generates, in the worst case, exponentially many trees. To deal with such the exponential blown up, Liang et al. [2013] consider only the L highest scoring candidates in each generation step.

Depending on a grammar, and the lexicon, semantic parsers build an unambiguous logical representation of the given sentence. For instance, "What is the birthplace of Barack Obama?" could be translated into $\lambda X : city(X) \wedge birth(Y, X) \wedge const(Y, BarackObama)$. The lexicon translates words or short phrases into a set of predicates, e.g. ($city$, $birth$, $const$), or set of entities such as ($BarackObama$). For instance, a textual 'birthplace' can be mapped into $\{city, birth\}$, and a textual 'Obama' can be mapped into $\{BarackObama, MichelleObama\}$. Liang et al. [2013] have proposed Dependency-Based Compositional Semantics to efficiently

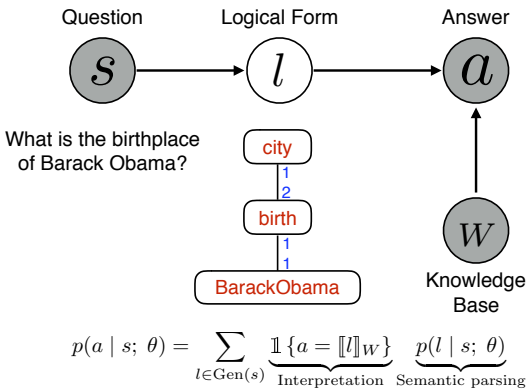


Figure 4.4: Probabilistic Graphical Model of the semantic parser proposed by Liang et al. [2013]. Grey colored circles denote observed variable, while the white colored circle denotes a latent variable. Latent variable is marginalized out for predictions. Logical form l is interpreted according to a Knowledge Base $\mathcal{W}$ to produce an answer a for the given question s.

encode logical formulas, in which each formula is represented as a tree with predicates as nodes and relations between the predicates as edges (Figure 4.3 shows a few examples of the Dependency-Based Compositional Semantics encoding). Edges encode constrains between the nodes. Numbers represent localization of bindings between the arguments. For example, the relation $1-2$ between *city* and *birth* (middle of Figure 4.3) binds the second argument of the predicate $birth(X, Y)$ with the first, and only one, argument of the predicate $city(X)$. Let $\left\langle p,\ {}^{j_1}_{i_1} : c_1,\ {}^{j_2}_{i_2} : c_2,\ ...,\ {}^{j_k}_{i_k} : c_k \right\rangle$ be the formal representation of a tree with a parent p and k children $c_1, ..., c_k$ that are related with the parent via a binding relation ${}^{j}_{i}$ each. For instance, $\langle city,\ {}^{1}_{2} : \langle birth,\ {}^{1}_{1} : BarackObama \rangle \rangle$ represents the middle tree in Figure 4.3. Then the interpretation of the tree can be defined recursively. Precisely, $[\![\left\langle p,\ {}^{j_1}_{i_1} : c_1,\ {}^{j_2}_{i_2} : c_2,\ ...,\ {}^{j_k}_{i_k} : c_k \right\rangle]\!]_{\mathcal{W}}$ equals the interpretation of the parent node intersected with interpretations of its children, that is $\{t \mid t \in [\![p]\!]_{\mathcal{W}}\} \cap \bigcap_{l=1}^{k} \{t \mid v_{i_l} = t_{j_l},\ v \in [\![c_l]\!]_{\mathcal{W}}\}$. Let us consider the denotation of the question "What is the birthplace of Barack Obama?" using Dependency-Based Compositional Semantics

$$
\begin{aligned}
[\![\text{What is the birthplace of Barack Obama?}]\!]_{\mathcal{W}} = \\
[\![\langle city,\ {}^{1}_{2} : \langle birth,\ {}^{1}_{1} : BarackObama \rangle \rangle]\!]_{\mathcal{W}} = \\
\{t \in [\![city]\!]_{\mathcal{W}}\} \cap \{t \mid v_2 = t_1,\ v \in [\![\langle birth,\ {}^{1}_{1} : BarackObama \rangle]\!]_{\mathcal{W}}\} = \\
\{t \in [\![city]\!]_{\mathcal{W}}\} \cap \{t \mid v_2 = t_1,\ v \in \{u \in [\![birth]\!]_{\mathcal{W}}\} \cap \{u \mid u_1 = BarackObama\}\} = \\
\{t \in [\![city]\!]_{\mathcal{W}}\} \cap \{t \mid v_2 = t_1,\ v \in \{u \mid birth(BarackObama, u_2)\}\} = \\
\{t \in [\![city]\!]_{\mathcal{W}}\} \cap \{t \mid birth(BarackObama, t_1)\} = \\
\{t \in [\![city]\!]_{\mathcal{W}}\} \cap \{t \mid birth(BarackObama, t)\} = \\
\{Honolulu\}
\end{aligned}
$$

Hence, the final denotation of the question "What is the birthplace of Barack Obama?" is a singleton {Honolulu}, the answer to the question. Such an encoding of the logical forms

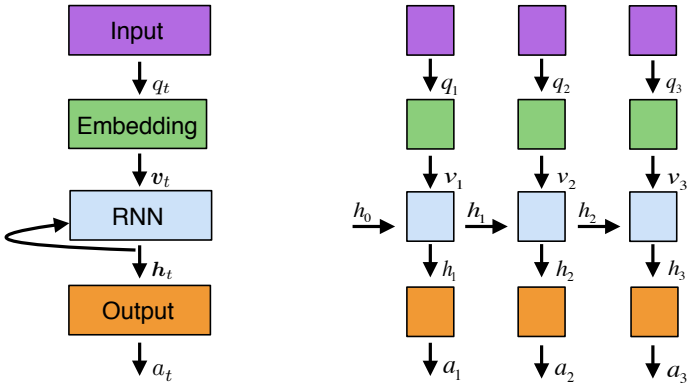


Figure 4.5: Two equivalent depictions of Recurrent Neural Networks are shown: rolled (left) and unrolled (right). As we can see, Recurrent Neural Networks repeatedly apply a non-linearity (blue box) to its input $\boldsymbol{v}_t$, and its previous hidden state $\boldsymbol{h}_{t-1}$. This is possible due to the weights sharing.

precisely leads to a valid denotation computed with the following complexity: number of all nodes times the largest interpretation of a node. The interpretation mechanism can be extended to work with other relations (e.g. aggregation shown in the most right tree in Figure 4.3). However, such a detailed exposition is beyond the scope of this thesis, and instead a curious reader is welcome to read Liang et al. [2013]. This semantic parser is further extended to handle a Visual Turing Test in Chapter 9.

Pros and cons The representation of the meaning can be as powerful as a formal language that we use to describe it. For instance, Dependency-Based Compositional Semantics can represent counting questions, or negations. Moreover, the semantic parser can also explain decisions made to derive an answer by showing its derivations in the formal language (e.g. Figure 4.3 show a few derivations). On the other hand, using semantic parsers require a hand-designed ontology. As we will see later in this thesis (Chapter 9 and Chapter 10), the dependency on an ontology and the actual content of the Knowledge Base can be a serious bottleneck of this approach to work within the visual domain. Interestingly, benefits and drawbacks of the next approach to represent natural language sentences that we discuss in this chapter are reversed.

4.3 Recurrent Neural Networks

Overall picture Semantic parsers have a few drawbacks. Most importantly, they have to deal with exponentially many ways of deriving the meaning, and require a hand-designed ontology. Hence, errors that are made in defining predicates strongly impact the overall performance of the architecture. This is especially problematic when dealing with images. Therefore, we seek alternative ways of representing a textual input. Recurrent Neural Networks are Deep Learning approaches to capture the dynamics of a sequential input. Important for us, they are jointly trained, end-to-end, and scalable architectures that can

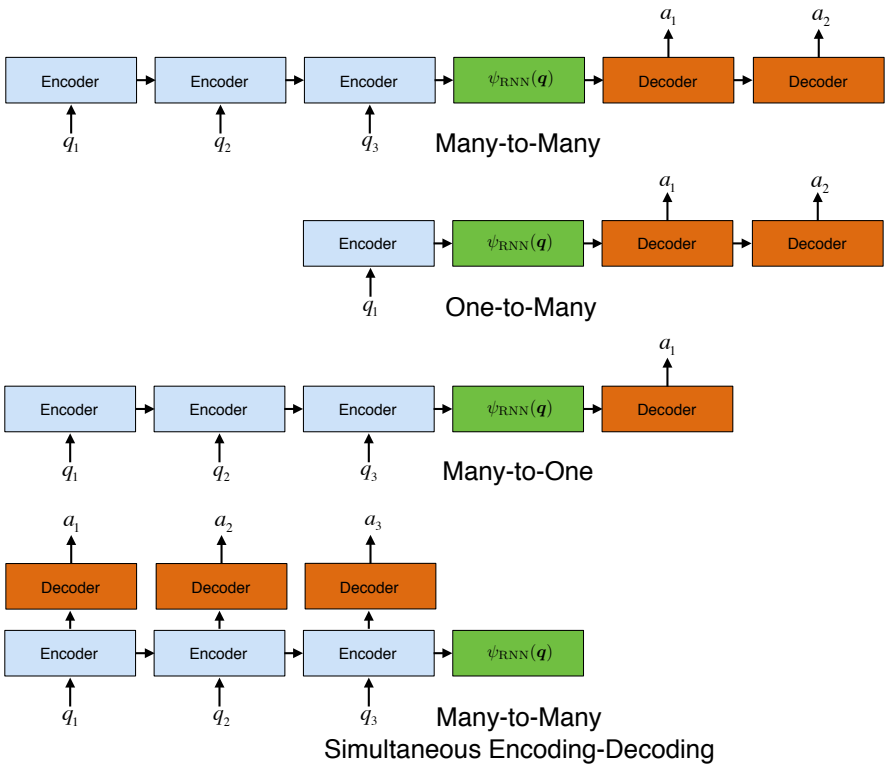


Figure 4.6: Various mapping problems – many-to-many, one-to-many, many-to-one – that can be handled with recurrent encoder-decoders. We can also see that the decoder can be applied directly after the encoder (at the bottom of the figure). The green box, $\phi_{\text{RNN}}(\boldsymbol{q})$ depicts the representation of a sequential input $\boldsymbol{q}$.

be trained for sequential tasks, e.g. various Natural Language Understanding tasks.

Historical outline Hopfield network [Hopfield 1982] is among the first Recurrent Neural Networks that is designed to recover a pattern from a corrupted input. Early discriminatively trained models are introduced by Jordan [1986], and Elman [1990]. The networks from these days were quite difficult to train due to the gradient vanishing problem, and therefore they could not represent longer-term dependencies. To handle such an issue, Hochreiter and Schmidhuber [1997] have proposed a gating mechanism that allows the 'history' to flow unchanged, to add the information from the current input, or to forget the 'history'. Such Recurrent Neural Networks are named Long-Short Term Memory Networks, or just LSTM. Later on numerous variants of LSTM have been proposed. For instance, Gated Recurrent Unit (GRU) [Cho et al. 2014] not only simplifies LSTM by reducing the number of gates from four to only two, but also maintains a competitive performance. In this thesis, we use LSTM and GRU. Therefore, now, we will explain both neural networks formally.

Outline of RNN Let $\boldsymbol{q}_{T_q}$ be an input sequence consisting of T_q input elements, i.e. $\boldsymbol{q}_{T_q} = (q_1, q_2, ..., q_{T_q})$. Let $\boldsymbol{a}_{T_a} = (a_1, a_2, ..., a_{T_a})$ be a target sequence consisting of T_a target elements. With such a notation, Recurrent Neural Networks are parameterized functions

that map source sequences onto target sequences, i.e. $F(\boldsymbol{q}_{T_q};\ \Theta) = \boldsymbol{a}_{T_a}$. Parameters are typically learnt in a supervised way based on the training set. If $T_q = 1$ and $T_a > 1$, then we have one-to-many mapping problem, for instance the image description task. If $T_q > 1$ and $T_a = 1$, then we have many-to-one mapping problem, for instance the video classification task. If $T_q > 1$ and $T_a > 1$, then we have many-to-many mapping problem, for instance the video description or question answering task. Recurrent Neural Networks maintain the 'history' by repeatedly applying a function to the current input and its previous hidden state. Formally, $\boldsymbol{h}_t := f(\boldsymbol{v}_t, \boldsymbol{h}_{t-1};\ \Theta)$, where $\boldsymbol{h}_t$ is a hidden state at time t, $\boldsymbol{v}_t$ is a vector-based representation (embedding) of the input element q_t, and Θ denotes all learnable parameters. For instance, Simple Recurrent Neural Networks set the update rule $\boldsymbol{h}_t := f(\boldsymbol{v}_t, \boldsymbol{h}_{t-1};\ \Theta)$ to be

$$\boldsymbol{h}_t := \sigma\left(W_{vh}\boldsymbol{v}_t + W_{hh}\boldsymbol{h}_{t-1} + \boldsymbol{b}\right) \tag{4.2}$$

where σ is some non-linearity. This is shown in Figure 4.5. As we can see, using of Recurrent Neural Networks is feasible due to the weights sharing, i.e. we keep the same weights at different time steps. The hidden state $\boldsymbol{h}_t$, which combines the input at the time step t with the hidden representation of the already observed subsequence, is a representation of the sequence $(q_1, q_2, ..., q_t)$. In this way, the final representation of the sentence $\boldsymbol{q}$ is $\psi_{\text{RNN}}(\boldsymbol{q}) := \boldsymbol{h}_{T_q}$. A decoder takes some sequence of hidden states, and decodes the target $\boldsymbol{a}$. For instance, in many-to-one mapping problems, the decoder is often a classifier that takes $\psi_{\text{RNN}}(\boldsymbol{q})$ and outputs a class label. In Chapter 10 and Chapter 11, we consider decoders that either use a Recurrent Neural Network or a classifier to decode the answer from $\psi_{\text{RNN}}(\boldsymbol{q}, \boldsymbol{x})$, where $\boldsymbol{q}$ is a question, and $\boldsymbol{x}$ is an image. Other possibilities are best explained by visualizations in Figure 4.6. Such architectures are jointly trained by backpropagation through time. For instance, 'simultaneous many-to-many' architectures (Figure 4.6) are often trained by minimizing the following cross-entropy loss

$$\arg\min_{\Theta} -\sum_{j}\sum_{t}\sum_{a_t} p(a_t \mid q_t^j)\log p_\Theta(a_t \mid q_t^j) \tag{4.3}$$

where q_t^j and a_t are respectively an input and target from the j-th data sample, and the t-th time step; $p(\cdot)$ is a data distribution, and $p_\Theta(\cdot)$ is the probability outputted by the Recurrent Neural Network, e.g. $p_\Theta(a \mid s) = \exp\left(W_{(a,:)}\text{RNN}(s;\Theta_{\text{RNN}})\right) / \sum_{\hat{a}} \exp\left(W_{(\hat{a},:)}\text{RNN}(s;\Theta_{\text{RNN}})\right)$, where $\text{RNN}(\cdot)$ is the recurrent encoder-decoder. Often $p(a_t \mid q_t^j) := \mathbb{1}\left\{a_t = a_t^j\right\}$ is used in Equation 4.3, with a_t^j as the j-th target sequence from the training set.

GRU Learning longer-term dependencies by Simple Recurrent Neural Networks, defined by Equation 4.2, is difficult [Hochreiter 1991; Bengio et al. 1994]. As a remedy, Hochreiter and Schmidhuber [1997] have proposed Long-Short Term Memory Networks (LSTM), and later on Cho et al. [2014] have introduced Gated Recurrent Unit (GRU). Both methods approach the problem of maintaining longer-term dependencies by a data-dependent gating mechanism. Here, we first explain GRU because it is a simpler and yet competitive network.

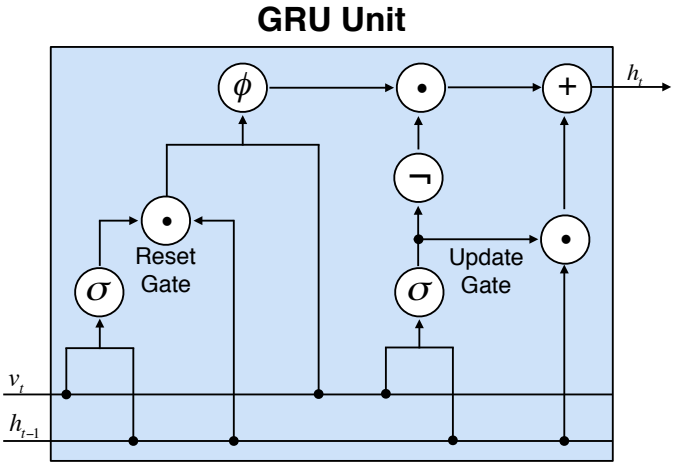


Figure 4.7: Gated Recurrent Unit. ¬ is the negation unit defined as 1 − input. Black dots symbolize connections.

GRU is described by the following set of equations:

$$\boldsymbol{r}_t = \sigma(W_{vr}\boldsymbol{v}_t + W_{hr}\boldsymbol{h}_{t-1} + \boldsymbol{b}_r) \tag{4.4}$$

$$\boldsymbol{u}_t = \sigma(W_{vu}\boldsymbol{v}_t + W_{hu}\boldsymbol{h}_{t-1} + \boldsymbol{b}_u) \tag{4.5}$$

$$\boldsymbol{c}_t = W_{vc}\boldsymbol{v}_t + W_{hc}(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}) + \boldsymbol{b}_c \tag{4.6}$$

$$\boldsymbol{h}_t = \boldsymbol{u}_t \odot \boldsymbol{h}_{t-1} + (\mathbf{1} - \boldsymbol{u}_t) \odot \phi(\boldsymbol{c}_t) \tag{4.7}$$

where σ is the sigmoid function, ϕ is the hyperbolic tangent, and $\boldsymbol{v}_t$, $\boldsymbol{h}_t$ are input and hidden states at the time step t; $\boldsymbol{r}$ and $\boldsymbol{u}$ are reset and update gates respectively, while $\boldsymbol{c}$ is a cell state; $\odot$ is a piecewise multiplication of two vectors. The role of the gates, composed of a sigmoid and a piecewise multiplication, is to decide how much of information should be passed through. Since $\sigma(\cdot) \in [0, 1]$, let us consider the extreme cases. If $\boldsymbol{r}_t = \mathbf{1}$ then the cell state $\boldsymbol{c}_t$ is influenced by both, the current input $\boldsymbol{v}_t$ and the previous hidden state $\boldsymbol{h}_{t-1}$. If $\boldsymbol{r}_t = \mathbf{0}$ then the cell state $\boldsymbol{c}_t$ is influenced only by the current input $\boldsymbol{v}_t$ (it is 'reseted'). Similarly, if $\boldsymbol{u} = \mathbf{1}$ then the current input $\boldsymbol{v}_t$ is completely ignored in the update of the hidden state $\boldsymbol{h}_t$. If $\boldsymbol{u} = \mathbf{0}$ then the hidden state is updated only based on the state cell $\boldsymbol{c}_t$. However, since $\sigma(\cdot) \in [0, 1]$ a some fraction of current and historical information is passed through than completely ignored. This network is depicted in Figure 4.7.

LSTM LSTM also uses the gating mechanism, but it is described by more equations:

$$\boldsymbol{i}_t = \sigma(W_{vi}\boldsymbol{v}_t + W_{hi}\boldsymbol{h}_{t-1} + \boldsymbol{b}_i) \tag{4.8}$$

$$\boldsymbol{f}_t = \sigma(W_{vf}\boldsymbol{v}_t + W_{hf}\boldsymbol{h}_{t-1} + \boldsymbol{b}_f) \tag{4.9}$$

$$\boldsymbol{o}_t = \sigma(W_{vo}\boldsymbol{v}_t + W_{ho}\boldsymbol{h}_{t-1} + \boldsymbol{b}_o) \tag{4.10}$$

$$\boldsymbol{g}_t = \phi(W_{vg}\boldsymbol{v}_t + W_{hg}\boldsymbol{h}_{t-1} + \boldsymbol{b}_g) \tag{4.11}$$

$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \boldsymbol{g}_t \tag{4.12}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \phi(\boldsymbol{c}_t) \tag{4.13}$$

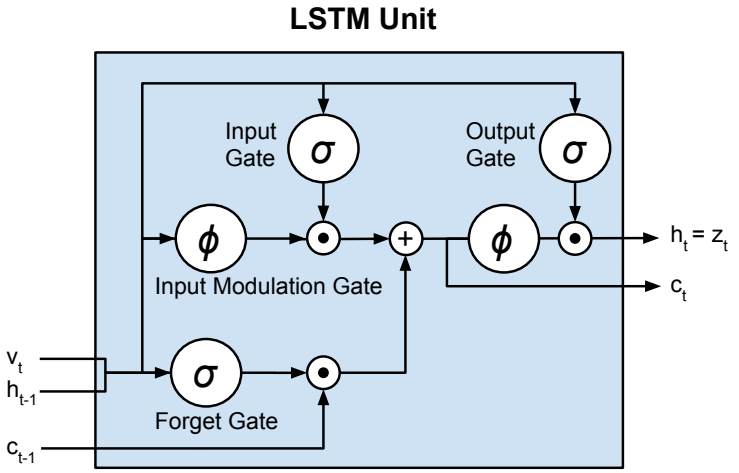


Figure 4.8: Long-Short Term Memory.

where σ is the sigmoid function, ϕ is the hyperbolic tangent, and $\boldsymbol{v}_t$, $\boldsymbol{h}_t$ are input and hidden states at time t; $\boldsymbol{c}_t$ represents memory cells; $\boldsymbol{i}_t$, $\boldsymbol{f}_t$, $\boldsymbol{o}_t$ are input, forget, and output states respectively; $\odot$ is a piecewise multiplication of two vectors. Arguably, the most important change to GRU is an output gate that decides how much of information should be outputted. A similar analysis, which for the sake of brevity we omit here, that we did before with GRU lead to a better understanding of the equations. The network is depicted in Figure 4.8.

Pros and cons Recurrent Neural Networks together with Convolutional Neural Networks, presented in Section 3.3, provide a compelling, unified view that inputs are mapped into fixed-length vector representations. Afterwards, we can use vector operations to combine the vectors together. For instance, in Chapter 11, we explore piece-wise summation, multiplication, and concatenation in order to combine two modalities, questions and images. Moreover, they are also scalable, jointly and end-to-end trained approaches to represent the meaning, which handles a few issues we encounter while working with semantic parsers. On the other hand, such networks are less transparent than semantic parsers, and learning logical operations still remain an open challenge.

4.4 Conclusion

In this thesis, we use two different approaches to language understanding, Recurrent Neural Networks and semantic parsers. Each such an approach has unique advantages. Semantic parsers offer good introspection into the 'behavior' of the algorithm, and can readily handle negations and quantifications. On the other hand, Recurrent Neural Networks are jointly trained, end-to-end, scalable approaches that currently lead in various variants of Visual Turing Test. On the other hand, they are less transparent, and learning logical operations such as negations or quantification has not been yet shown.

Chapter 5

Related Work

Contents

In this thesis, we pursue three themes: spatial pooling for visual recognition, spatial relations for text-to-image retrieval, and question answering about images (Visual Turing Test). This chapter focuses on prior, contemporaneous, and subsequent work that tightly relates to the thesis, while the following chapters discuss only prior work specific to the respective chapter at the time of publishing the corresponding work.

5.1 Spatial Pooling Regions

The first theme of the thesis is concerned with spatial pooling regions, in which we contribute by a generalization of a pooling operator that allows to discriminatively learn a spatial layout. Our work was presented at ICLR'13 Workshop, and BMVC'13, and is described in Chapter 6. Consequently, in Section 5.1.1, we discuss various approaches that build upon the spatial pooling regions idea and precede our work. Next, in Section 5.1.2, we connect our work with other approaches, to either devise spatial pooling schemes or generalize a pooling operator, which have appeared after the publication of our work.

5.1.1 Prior Work

Around the year 2011 the leading image classification methods were built following the Spatial Pyramid Matching (SPM) framework [Lazebnik et al. 2006; Yang et al. 2009], which uses a hierarchical representation to built image features. However, each level in such a hierarchy has been either hand-designed, or independently trained. Pooling over spatial regions play a

particularly important role in this framework. A few researchers have, however, questioned the arbitrariness of the spatial division used in the original architecture [Lazebnik et al. 2006; Yang et al. 2009], and proposed alternative approaches to derive spatial layouts [Jia and Huang 2011; Sharma and Jurie 2011; Jia et al. 2012; Russakovsky et al. 2012; Sánchez et al. 2012; Feng et al. 2011; Krapac et al. 2011; Koniusz and Mikolajczyk 2011]. Jia and Huang [2011] optimizes binary pooling strategies that are given by the superposition of the rectangular basis. Spatial regions of Sharma and Jurie [2011] are also formed of rectangular receptive fields, but obtained by successive splittings of the cell, and are not interpreted as basis. Russakovsky et al. [2012], and Sánchez et al. [2012] have shown improvement over SPM by pooling the objects and background separately. Krapac et al. [2011] and Koniusz and Mikolajczyk [2011] model spatial location of the visual words by fitting Mixture of Gaussians. In contrast to the aforementioned approaches, our method [Malinowski and Fritz 2013b,a] discriminatively learns shapes of the pooling regions without resorting to the notion of the bounding boxes, or any other geometrical primitives. Moreover, we train the spatial layout, which we believe could also be interpreted as an image-independent variant of an attention mask [Xu et al. 2015] that has recently gained popularity in the Computer Vision community.

5.1.2 Contemporaneous and Subsequent Work

In our work, we allow for a discriminative training of a spatial pooling layout by a suitable generalization of a pooling operator. Most of the contemporaneous and subsequent work build upon one of the following directions: either by developing alternative approaches to derive spatial layouts, or by proposing alternative ways to generalize a pooling operator. In the remaining of this section, we present work belonging to such directions.

Spatial pooling layout Task-dependent spatial regions have also been used in subsequent works as follows. Eweiwi et al. [2015] argues for more global spatial regions obtained through applying a Non-negative Matrix Factorization technique to a motion flow for an action recognition task. Wang and Tan [2016] use a more compact encoding with smaller receptive fields of the pooling operator to create an image representation. Their argument about the importance of finer-grained pooling operation while low-dimensional encoding scheme is used is consistent with our findings. The work of Liu et al. [2016] is tightly connected to our learnable pooling regions framework. Similarly to us, they also argue for a discriminatively, and jointly trained spatial regions together with a classifier. However, in addition, they learn category-specific pooling regions with a boosting-like technique to improve the efficiency of training.

Generalizing a pooling operator We achieve our goals of obtaining a more efficient spatial layout by a suitable generalization of the pooling operator. Various ways of generalizing this operator have also been a research topic of the Deep Learning community. For instance, Zeiler and Fergus [2013] introduce a stochastic variant of the sum-pooling operator, where neural activations are multiplied by probabilities. Yu et al. [2014] suggest to randomly sample either max-pooling or sum-pooling operator in the pooling stage. Lee et al. [2016]

have further extended the previous idea by introducing a mixed (sum, max), gated, and tree pooling operator. Goodfellow et al. [2013] introduce Maxout pooling, where max-pooling operation is applied to a set of linearly transformed activations. A bit closer to the main idea of our work, Lebedev and Lempitsky [2015] train a Convolutional Neural Network to learn a spatial layout of the receptive fields of the network by enforcing group-sparsity constraints. He et al. [2016] use a data-dependent pooling scheme, where the features from Fully Convolutional Network are pooled together according to the predicted segmentation mask, so that features belonging to the same segment are grouped together. Finally, Li et al. [2015] have proposed a theoretical framework to explain the pooling operation, and base on the theoretical findings they have developed a multi-scale, multi-domain pooling pipeline.

5.2 Spatial Relations and Retrieval

The second theme of the thesis is concerned with spatial reasoning, in which we contribute by proposing spatial templates used in spatial reasoning as well as a compositional architecture for the text-to-image retrieval task. This line of research was summarized in Malinowski and Fritz [2014c], and is described in Chapter 7, and Appendix A. Consequently, in Section 5.2.1, we discuss prior work from various research fields that we directly drew inspirations from. Next, in Section 5.2.2, we show how other, contemporaneous and subsequent, work is related to our work on spatial relations.

5.2.1 Prior Work

In this line of research, we are mostly interested in data-driven approaches to model spatial relations for the text-to-image retrieval task. In our work, we argue that text-to-image retrieval architectures should ground spatial relations to handle the image-sentence alignment problem. Moreover, we also show how to leverage our work on the spatial pooling regions to build spatial templates that are next used to reason about a spatial relationship between objects.

Modeling spatial relations This part of the thesis is mainly influenced by the work of Lan et al. [2012]. In that work, the authors address a problem of the text-to-image retrieval task with structured queries, in which a textual input with a binary spatial preposition between the nouns. Our work goes beyond the structured queries and a restricted spatial vocabulary of the work of Lan et al. [2012]. For the purpose of scaling up such work to real-world scenarios, instead of using a hand-designed representation of a few relations ('above', 'below', 'overlap' like in Lan et al. [2012]), we propose a flexible and learnable representation that is based on spatial templates used in some psychological studies [Logan and Sadler 1996], and thus can be interpreted as a version of the learnable pooling regions [Malinowski and Fritz 2013a,b] centered at the reference object. A similar idea of using spatial templates is also proposed by Fritz et al. [2007] and also used in Fritz et al. [2010]. However, in contrast to our approach, Fritz et al. [2007] and Fritz et al. [2010] hand-design a set of four such templates to model 'left of', 'right of', 'above', and 'below'.

Grounding spatial relations Although research on grounding of spatial language has a long standing tradition, previous methods mostly focus on rule-based spatial representation [Moratz and Tenbrink 2006; Kruijff et al. 2007] or more recently on a set of hand-crafted spatial features with learnt weights [Tellex et al. 2010; Golland et al. 2010; Lan et al. 2012; Guadarrama et al. 2013b]. Although the latter approaches show improvements they still rely on designing the right set of features and their generalization and scalability to many spatial relations have not been proven yet. For instance, Lan et al. [2012] use only 2 spatial prepositions.

Image-sentence alignment Successful approaches to handle the text-to-image retrieval task need to align sentences with the corresponding images [Lin et al. 2014a; Kong et al. 2014]. Recent research on embedding [Socher et al. 2014; Karpathy et al. 2014; Mao et al. 2014] have opened a door for bi-directional methods that retrieve images based on a textual input, or sentences from a given image. However, in contrast to our work, none of these methods use spatial reasoning to improve the alignment. Karpathy et al. [2014] learns an embedding between textual and visual fragments, which becomes particularly attractive to us as we seek spatial relationships between the pairs of fragments.

Spatial pooling regions Spatial pooling has been proven to work well in many recognition tasks [Lazebnik et al. 2006; Yang et al. 2009], and is still a part of many recent approaches [Krizhevsky et al. 2012]. Although the research literature is densely populated with many variations of the spatial pooling regions framework, to the best of our knowledge there is no work that links pooling regions with spatial reasoning on object detections in a scene. In this work, we fill this gap and show a suitable interpretation of the framework. Closely related to our work is an object-centric pooling [Russakovsky et al. 2012] that relies on the object localization methods to distinguish between a foreground and background and next pool over both regions separately. Although, our method is also based on the localization of different objects, we spatially relate every pair of detections in the image to reason about their spatial arrangement.

5.2.2 Contemporaneous and Subsequent Work

Spatial relations are further investigated in a few subsequent work that we enlist in this section. Parser-based compositionality that induces a topology of a neural network, based on which we build our architecture to deal with spatial relations (Appendix A), turned out to be investigated, in parallel to our work, in the NLP community, and later also presented to the Visual Turing Test community.

Spatial relations Christie et al. [2016] propose an approach to simultaneously perform semantic segmentation and prepositional phrase attachment resolution. They argue that some language ambiguities can only be resolved together with the corresponding image. Although a spatial reasoning plays a role in the proposed method, as opposite to our approach, it is more implicit. Malinowski and Fritz [2014a] show the importance of spatial relations for the 'question answering about images' task. However, the proposed method,

as opposite to our approach to reason about spatial relations, uses hand-designed set of rules for the spatial resolution. The work of Andreas et al. [2016b] define neural modules to handle the question answering about images task. For instance, the neural module *Transform* shifts attention of the network according to the spatial preposition. Similarly to our method, Andreas et al. [2016b] do not define a set of spatial rules, but rather rely on learning them from data.

Compositional Neural Networks Our Data-Driven Compositional Neural Architectures (DDCNA is presented in Appendix A, but also used in Chapter 7) resemble Recursive Neural Networks introduced by Socher et al. [2013], in which a topology of the network depends on the output of a syntactic parser. Similarly, Andreas et al. [2016b] also propose a neural network that topology is input-dependent.

5.3 Towards a Visual Turing Test

The third theme of the thesis is concerned with the Visual Turing Test– a holistic task in which machines have to answer on a series of questions about the content of real-world images. We have contributed to this field in numerous ways: we are first who propose a dataset about question answering about real-world images, present first methods to handle the task, and extend the accuracy metric to handle ambiguities in the answers. Various aspects of this work were presented at NIPS'14 [Malinowski and Fritz 2014a], NIPS'14 Workshop [Malinowski and Fritz 2014b], AAAI'15 Workshop [Malinowski and Fritz 2015], ICCV'15 [Malinowski et al. 2015], ICMR'16 [Chowdhury et al. 2016a] , and BMVC'16 [Mokarian et al. 2016]. This line of research is also described in Chapter 8, Chapter 9, Chapter 10, and Chapter 11. Consequently, in Section 5.3.1, we discuss numerous work from various fields that directly inspired us to pursue the Visual Turing Test. Next, in Section 5.3.2, we discuss work that follow up our research on the Visual Turing Test.

5.3.1 Prior Work

As we argue in Chapter 8, answering questions about the content of images requires taking a holistic view, and therefore various techniques from Computer Vision, Natural Language Understanding, and Machine Learning should be employed. These techniques include: a representation of a visual input (Section 5.3.1.1), a representation of a textual question (Section 5.3.1.2), and a multimodal fusion (Section 5.3.1.4). 'Textual question answering' is a sister field to the Visual Turing Test that we take some inspirations from (Section 5.3.1.5). In our endeavor to build a holistic machine that answer questions, we are also influenced by various techniques proposed to ground language in the visual world (Section 5.3.1.6).

5.3.1.1 Encoding a Visual Input

Extracting a good representation from the visual content is an important component in developing approaches towards the Visual Turing Test. Since the proposal of AlexNet [Krizhevsky et al. 2012], Convolutional Neural Networks (CNNs) have become dominant and

most successful approaches to build a vector representation of an image. CNNs directly learn the representation from the raw image data and are trained on large image corpora, typically ImageNet [Russakovsky et al. 2014]. Interestingly, after these models are pre-trained on ImageNet, they can typically be adapted to other tasks. In this thesis, among the other things, we evaluate how well the most dominant and successful CNN models can be adapted to the Visual Turing Test. Specifically, we evaluate *AlexNet* [Krizhevsky et al. 2012], *VGG* [Simonyan and Zisserman 2015], *GoogleNet* [Szegedy et al. 2015], and *ResNet* [He et al. 2015]. These models, reportedly, achieve increasingly better accuracies on the ImageNet dataset, and hence, arguably, serve as stronger models of the visual perception.

5.3.1.2 Encoding a Textual Input

The other important component to answer a question about an image is to understand the natural language question, which means here building a representation of a variable length sequence of words (or characters, but we will focus only on the words in this work). In this thesis, among the other things, we experiment with numerous approaches to represent the language, starting from Semantic Parsers [Liang et al. 2013] (Section 5.3.1.3), through Bag-Of-Words of words embeddings that ignore an order in the sequence of words, ending with Convolutional Neural Networks [Kim 2014; Kalchbrenner et al. 2014] and Recurrent Neural Networks [Hochreiter and Schmidhuber 1997; Cho et al. 2014; Sutskever et al. 2014].

5.3.1.3 Semantic Parsers

We build our first approach to answer questions about images based on a semantic parser (concretely, we use it in Chapter 9 to build a logic-based approach; in Chapter 4 we briefly introduce a semantic parser to the curious reader). In the following, we enumerate developments of the semantic parsers, ranging from the first rule-based parsers, to parsers that are induced from logical forms or denotations.

Rule-based Semantic Parses in the early seventies are rule-based, i.e. they follow a set of hand-designed templates that transforms a textual input into a formal representation readily accessible to a machine. Using this approach to interface between textual and formal representations of the language led to a few projects that gained a particularly high attention. Examples include STUDENT [Bobrow 1964] for solving high school algebra problems, and LUNAR [Woods 1978] for natural language interface into database with moon rocks. Arguably, the most famous project from that era is SHRDLU, which operators on a 'block world' with the goal of rearranging blocks in a certain way through natural language instructions [Winograd 1972]. Unfortunately, scaling up such approaches to other, larger domains become prohibitively difficult, and therefore other approaches to language understanding have gained popularity.

Induced from logical forms The next generation of Semantic Parsers is trained from the pairs of the form (*textual question*, *logical representation*) [Zelle and Mooney 1996; Zettlemoyer and Collins 2007, 2009; Wong and Mooney 2006; Kwiatkowski et al. 2010]. Although such parsers are more flexible than the rule-based ones, yet an expensive annotation

effort of the form of annotating textual questions with their logical representations make them still difficult to scale to other, larger domains.

Induced from denotations Most recently, parsers are induced from denotations, i.e. from the pairs of the form (*textual question*, *textual answer*) [Liang et al. 2013; Berant et al. 2013; Berant and Liang 2014]. This move has led to a relatively easy annotation process, and hence made it easier to scale the parser to other domains. In this thesis, we extend a parser from this generation to work with uncertainty in the representation of the visual world.

5.3.1.4 Combining RNNs and CNNs

The task of describing a visual content like still images as well as videos has been successfully addressed with a combination of encoding the image with CNNs and decoding, i.e. predicting the sentence description with an RNN [Donahue et al. 2015; Karpathy and Fei-Fei 2015; Venugopalan et al. 2015b; Vinyals et al. 2014; Zitnick et al. 2013]. This is achieved by using the RNN model that first gets to observe the visual content and is trained to afterwards predict a sequence of words that is a description of the visual content. In the thesis, we extend this idea to the Visual Turing Test, where we formulate a model trained to either generate or classify an answer based on the visual and natural language inputs.

5.3.1.5 Textual Question Answering

Answering on purely textual questions has been studied in the NLP community [Berant and Liang 2014; Liang et al. 2013] and state-of-the-art techniques typically employ semantic parsing to arrive at a logical form capturing the intended meaning and infer relevant answers. Only recently, the success of the previously mentioned Recurrent Neural Networks, has carried over to this task [Iyyer et al. 2014; Weston et al. 2014]. In the thesis, we stand on the shoulders of both research threads, and build a visual question answering architecture.

5.3.1.6 Grounding Language in the Visual World

SHRDLU [Winograd 1972] is among the first models that connect the language with an external world, in this case with so called 'block world'. More recent approaches to ground the language in the visual world are presented by Fritz et al. [2007], Fritz et al. [2010], Matuszek et al. [2012] and Krishnamurthy and Kollar [2013]. All the aforementioned approaches use images as the representation of the physical world, but concentrate rather on constrained domains with images consisting of very few objects, and rather simple interactions. For instance, Krishnamurthy and Kollar [2013] consider only two mugs, a monitor and a table in their dataset, whereas Matuszek et al. [2012] examine objects such as blocks, and building bricks. In a related research direction, Tellex et al. [2011] consider commands grounding for robotic navigation. In contrast, in this part of the thesis, we focus on a diverse collection of real-world indoor RGBD images [Silberman et al. 2012] - with many more objects in the scene and more complex spatial relationship between them. Moreover, our work on Visual Turing Test considers complex questions - beyond the scope of Matuszek et al. [2012] and

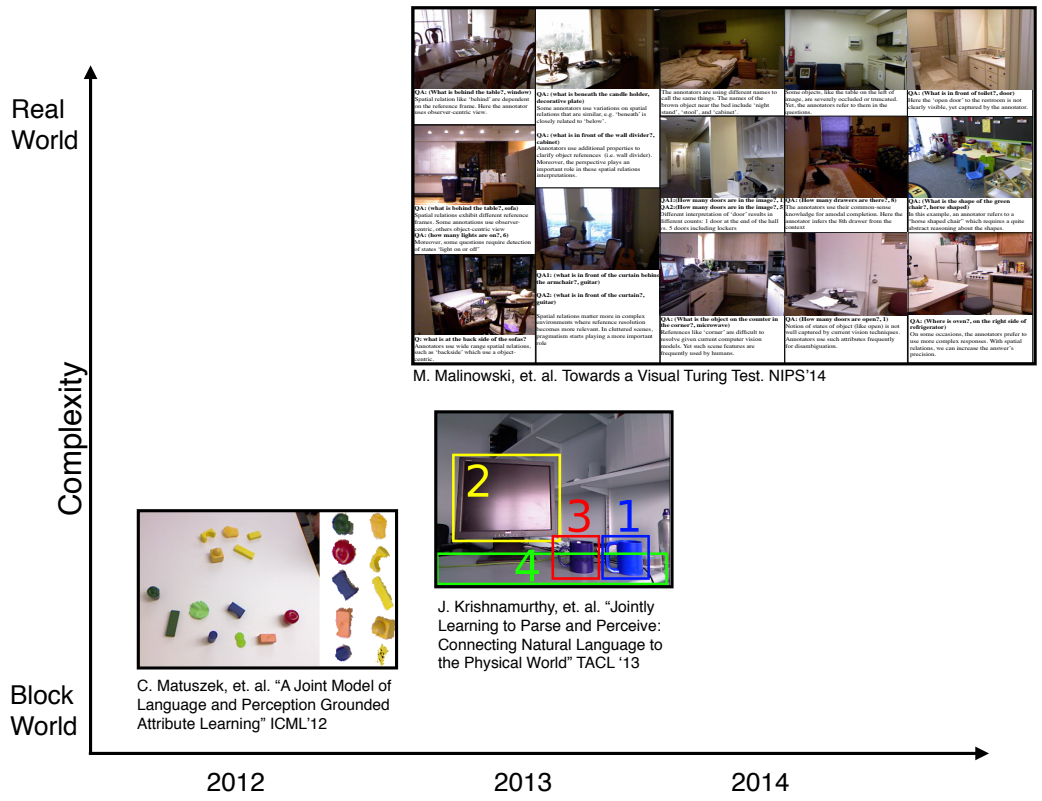


Figure 5.1: Various grounding tasks have been proposed over the years (x-axis). Y-axis indicates the complexity of the task. Early tasks, such as SHRDLU [Winograd 1972], consider 'block-world', while over the years tasks that better and better reflect real-world have been proposed. This thesis concerns with our project named Visual Turing Test (NIPS'14) [Malinowski and Fritz 2014a] that is inspired from the work on the grounding tasks. In this figure, we only show work on grounding that directly influenced us to pursuit the Visual Turing Test.

Krishnamurthy and Kollar [2013] - and reasoning across different images using only textual question-answer pairs for training. This imposes additional challenges for the question-answering engines such as scalability, good scene representation, dealing with uncertainty in the language and perception, efficient inference and spatial reasoning. Figure 5.1 shows the evolution of the tasks over the years, along with the increase of the complexity of each task.

5.3.2 Contemporaneous and Subsequent Work

Since our proposal of Visual Turing Test, and the first dataset that implements it [Malinowski and Fritz 2014a] together with the first methods and metrics, a few research labs have followed up the work. This section presents some subsequent work that extends the Visual Turing Test with new datasets, methods, or new insights. Figures 5.2, 5.3, and 5.1 show the subsequent datasets, models, and tasks respectively.

5.3.2.1 Datasets

Datasets are a driving force for the recent progress in the Visual Turing Test, recently also referred to as 'Visual Question Answering'. We have proposed the first Visual Turing Test dataset, which we call DAQUAR [Malinowski and Fritz 2014a]. The dataset has been later extended by including multiple human answers per question [Malinowski et al. 2015].

In parallel to DAQUAR, Geman et al. [2015] developed another variant of the Visual Turing Test. Their work, however, focuses on yes/no type of questions, provide detailed object-scene annotations, and does not require understanding of natural language. A large number of datasets that have appeared since our DAQUAR are summarized in Figure 5.2. Shortly after the introduction of DAQUAR, three other large-scale datasets have been proposed. All are based on the MS-COCO dataset of images [Lin et al. 2014b]. Gao et al. [2015] have annotated about $158k$ images with $316k$ Chinese question answer pairs together with their corresponding English translations. Ren et al. [2015a] have taken advantage of the existing annotations for the purpose of image description generation task and transformed them into question-answer pairs with the help of a set of hand-designed rules and a syntactic parser [Klein and Manning 2003]. This procedure has approximately generated $118k$ question-answer pairs. Finally, arguably nowadays the most popular, large scale dataset on question answering about images is VQA [Antol et al. 2015]. It has approximately $614k$ questions about the visual content of about $205k$ real-world images. Similarly to our Consensus idea (shown in Chapter 10 and Malinowski et al. [2015]), VQA provides 10 answers per each image. VQA has also about $150k$ questions about the abstract scenes. In this thesis, among the other things, we perform an experimental analysis on the VQA dataset and examine different variants of our neural-based method.

Although simple, automatic performance evaluation metrics have been a part of building first visual question answering datasets [Malinowski and Fritz 2014a,b, 2015], Yu et al. [2015b] have simplified the evaluation even further by introducing Visual Madlibs - a multiple choice question answering by filling the blanks task. In this task, a question answering architecture has to choose one out of four provided answers for a given image and the prompt. Formulating question answering task in this way has wiped out ambiguities in answers, and just a simple accuracy metric can be used to evaluate different architectures on this task. Yet, the task requires a holistic reasoning about the images, and despite of simple evaluation, it remains challenging for machines.

The Visual7W [Zhu et al. 2016] extends canonical question and answer pairs with additional groundings of all objects appearing in the questions and answers of an image by annotating the correspondences. It contains not only natural language answers, but also answers which require to locate the object (so called 'pointing questions'), which is then similar to the task of explicit grounding. In contrast to others such as VQA [Antol et al. 2015] or DAQUAR [Malinowski and Fritz 2014a] that has collected unconstrained question, the Visual7W focuses on the six, so called, Ws: what, where, when, who, why, and how, which can be answered with a natural language answer. An additional 7th question – which – requires a bounding box location as the answer. Similarly to Visual Madlibs [Yu et al. 2015b], Visual7W contains multiple-choice answers.

Related to Visual Turing Test, Chowdhury et al. [2016a] have proposed collective

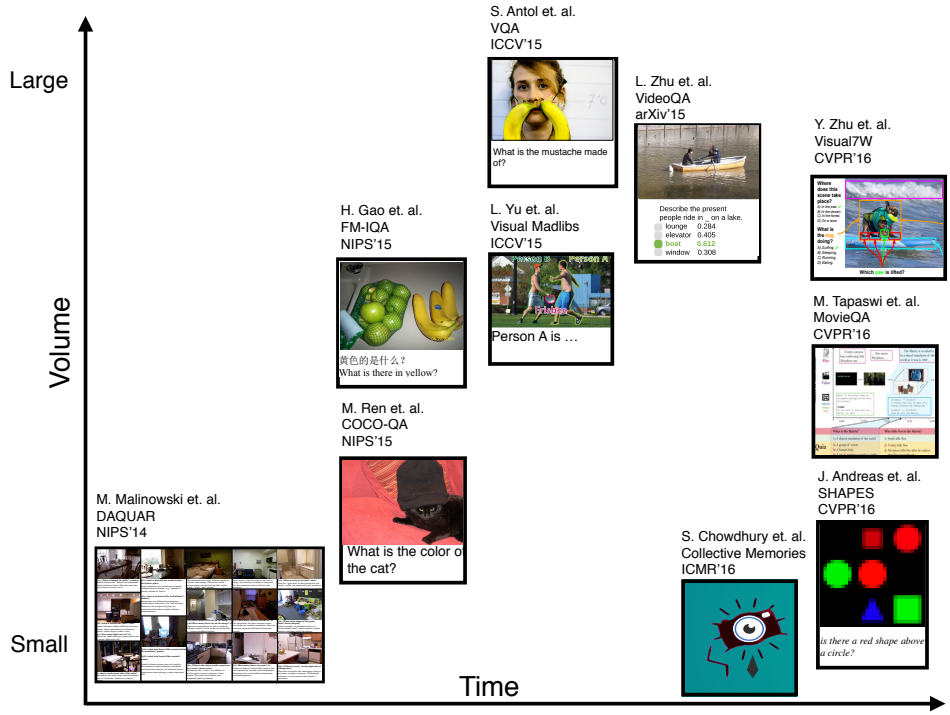


Figure 5.2: Since our DAQUAR (NIPS'14, bottom-left of the figure), numerous datasets have appeared. In this figure we characterize datasets according to their size (y-axis), and the time of appearance (x-axis). The volume is mostly influenced by the number of questions. Note that two datasets, MovieQA and VideoQA, have videos instead of images, and hence it is more difficult to properly determine their size with respect to other, image questions answering datasets. The depiction is symbolic.

memories and Xplore-M-Ego - a dataset of images with natural language queries, and a media retrieval system, in which 'answers' are images. This work focuses on a user centric, dynamic scenario, where the provided answers are conditioned not only on questions but also on the geographical position of the questioner.

While the majority of datasets include real-world images, Andreas et al. [2016b] have proposed the SHAPE dataset, which consists of synthetic images of abstract shapes appearing in various configurations, and compositional questions. Such a dataset allows to study some phenomena such as spatial relations or compositionality in isolation.

Finally, moving from asking questions about images to questions about videos enhances typical questions with a temporal structure. Zhu et al. [2015] propose a task which requires to fill in blanks the captions associated with videos. The task requires inferring the past, describing the present and predicting the future in a diverse set of video description data ranging from cooking videos [Regneri et al. 2013] over web videos [Trecvid 2014] to movies [Rohrbach et al. 2015b]. Tapaswi et al. [2016] propose MovieQA, which requires to understand long term connections in the plot of the movie. Given the difficulty of the data, both works provide multiple-choice answers.

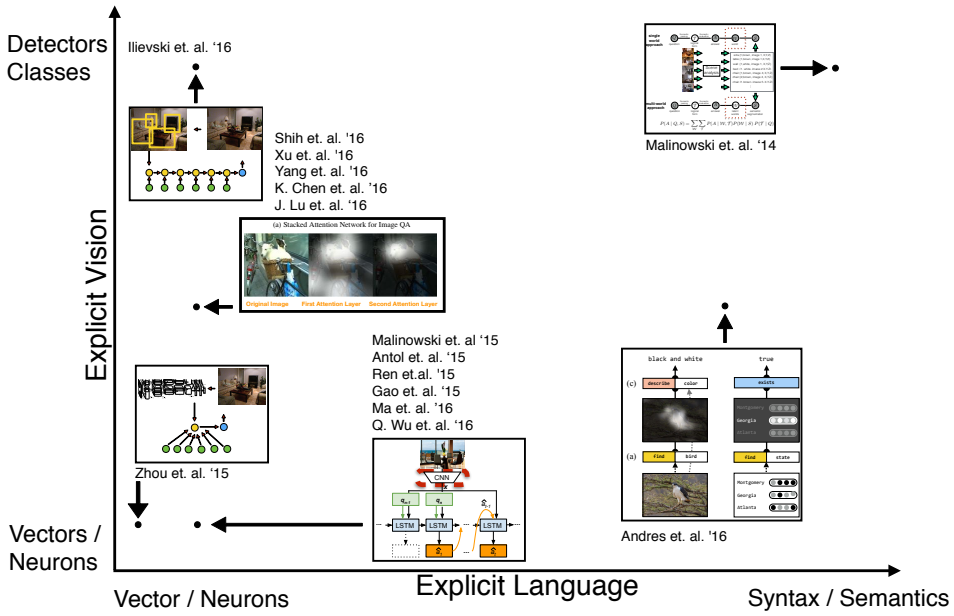


Figure 5.3: Since our first, logic-based approach to Visual Turing Test (NIPS'14, top-right of the figure), numerous methods have appeared. Similarly, many methods have followed our neural-based method (ICCV'15, bottom of the figure). In the x-axis, we characterize methods according to explicit assumptions they make about the structure in the language. In the y-axis, we characterize methods according to explicit assumptions they make about the structure in the vision.

5.3.2.2 Methods

Recently, a large number of architectures have been proposed to approach Visual Turing Test [Malinowski and Fritz 2014b]. They range from symbolic to neural-based approaches. There are also architectures that combine both symbolic and neural paradigms together. Some approaches use explicit visual representation in the form of bounding boxes surrounding objects of interest, while other use global full frame image representation, or soft attention mechanism. Yet others use an external knowledge base that helps in answering questions. Figure 5.3 depicts various methods depending on explicit/implicit assumptions they make while modeling a vision or language.

Symbolic-based approaches In our first work on the Visual Turing Test [Malinowski and Fritz 2014a], also presented in Chapter 9, we present a question answering system that is based on a semantic parser on a varied set of human question-answer pairs. Although it is the first attempt to handle question answering on DAQUAR, and despite its introspective benefits, it is a rule-based approach that requires a careful schema crafting, is not that scalable, and finally it strongly depends on the output of visual analysis methods as joint training in this model is not yet possible. Due to such limitations, the community has rather shifted towards either neural-based or combined approaches. This method is depicted in the top-right of Figure 5.3 since the semantic parser that the method uses requires an explicit,

compositional grammatical rules. Similarly, the method takes advantage of the bounding boxes, explicitly pointing to the objects in the image.

Neural-based approaches with full frame CNN Most contemporary approaches use a global image representation, i.e. they encode the whole image with a CNN. Questions are then encoded with an RNN [Malinowski et al. 2015; Ren et al. 2015a; Gao et al. 2015] or a CNN [Ma et al. 2015]. In contrast to symbolic-based approaches, neural-based architectures offer scalable and joint end-to-end training that liberates them from ontological commitment that would otherwise be introduced by a semantic parser. Moreover, such approaches are not 'hard' conditioned on the visual input and therefore can naturally take advantage of different language biases in question-answer pairs, which we interpret as learning common sense knowledge. Our work, presented in the thesis in Chapter 10 and Chapter 11, belong to this category [Malinowski et al. 2015, 2016], and is among the very first methods of this kind. These approaches to the Visual Turing Test are placed in the bottom-left of Figure 5.3 as they leave decisions about grammar, word and image representations to the neural network.

Neural-based approaches with an enhanced visual representation Global image representation can be enriched by using additional visual cues. For instance, Mokarian Forooshani et al. [2016] show that a better performance on the Visual Madlibs task can be achieved with a representation extracted from a large number of highly overlapping object proposals. They argue that such a modeling of the image builds a multi-part, and multi-scale image representation. A similar idea is also introduced by Tommasi et al. [2016], but instead of using a large number of proposals, they use a few, different detectors. Finally, they build a rich visual representation by integrating together different detectors as well as particular global representations that come from specialized CNNs.

Attention-based approaches Following Xu et al. [2015], who propose to use spatial attention for image description, Yang et al. [2015]; Xu and Saenko [2015]; Zhu et al. [2016]; Chen et al. [2015]; Shih et al. [2016]; Lu et al. [2016]; Fukui et al. [2016] predict a latent weighting (attention) of spatially localized images features (typically a convolutional layer of the CNN) based on the question. The weighted image representation rather than the full frame feature representation is then used as a basis for answering the question. In contrast to the previous models using attention, Dynamic Memory Networks (DMN) [Kumar et al. 2016; Xiong et al. 2016] first pass all spatial image features through a bi-directional GRU that captures spatial information from the neighboring image patches, and next retrieve an answer from a recurrent attention based neural network that allows to focus only on a subset of the visual features extracted in the first pass. Another interesting direction has been taken by Ilievski et al. [2016] who run state-of-the-art object detector of the classes extracted from the key words in the question. In contrast to other attention mechanisms, such an approach offers a focused, question dependent, 'hard' attention.

Answering with an external knowledge base Wu et al. [2016b] argue for an approach to the Visual Turing Test that first represents an image as an intermediate semantic attribute

representation, and next query external knowledge sources based on the most prominent attributes and relate them to the question. With the help of such an external knowledge base, such an approach captures a richer semantic representation of the world, beyond what is directly contained in images.

Compositional approaches A different direction is taken by Andreas et al. [2016b] who predict the most important components to answer the question with a natural language parser. The components are then mapped to neural modules, which are composed to a deep neural network based on the parse tree. While each question induces a different network, the modules are trained jointly across questions. This work compares to Malinowski and Fritz [2014a] by exploiting explicit assumptions about the compositionality of natural language sentences. Related to the Visual Turing Test, Malinowski and Fritz [2014c] have also combined a neural based representation with the compositionality of the language for the text-to-image retrieval task (Appendix A contains a more detailed exposition of the architecture).

Dynamic parameters Noh et al. [2015b] have an image recognition network and a Recurrent Neural Network (GRU) that dynamically change the parameters (weights) of a visual representation based on the question. More precisely, the parameters of its second last layer are dynamically predicted from the question encoder network and in this way changing for each question. While question encoding and image encoding are pre-trained, the network learns parameter prediction only from image-question-answer triples.

5.4 Concluding Remarks

In this thesis, we present our work that span three different Computer Vision areas: image recognition, text-to-image retrieval, and Visual Turing Test. We have built our work based on a large number of methods developed in various scientific fields, with the most tightly related prior work enlisted in this chapter (and with the following chapters of this thesis discussing only prior work specific to the respective chapter at the time of its publication). Two such scientific fields, Machine Recognition and Natural Language Understanding, are also introduced in Chapter 3 and Chapter 4. We also believe that our work has impacted and inspired the research community in numerous ways. In particular, with the rich subsequent work as the evidence, our approach to Visual Turing Test has greatly influenced the field. Appearance of a large number of various datasets that model different aspects of Visual Turing Test along with the methods to handle visual question answering is crucial to foster the progress on the holistic scene understanding, and therefore also building more advanced, 'intelligent' machines.

Chapter 6

Learning Smooth Pooling Regions for Visual Recognition

Contents

Visual recognition sits in the core of the Computer Vision research (Chapter 3), and is a key component in developing holistic machines (Chapter 1). In this chapter, we introduce a framework to learn smooth pooling regions. Later on, in Chapter 7, we show a link between spatial pooling regions and spatial templates that can be used in the text-to-image retrieval task.

From the early HMAX model to Spatial Pyramid Matching, spatial pooling has played an important role in visual recognition pipelines. By aggregating local statistics, it equips the recognition pipelines with a certain degree of robustness to translation and deformation yet preserving spatial information. Despite of its predominance in current recognition systems, we have seen little progress to fully adapt the pooling strategy to the task at hand. In this paper, we propose a flexible parameterization of the spatial pooling step and learn the pooling regions together with the classifier. We investigate a smoothness regularization term that in conjuncture with an efficient learning scheme makes learning scalable. Our framework can work with both popular pooling operators: sum-pooling and max-pooling. Finally, we show benefits of our approach for object recognition tasks based on visual words and higher level event recognition tasks based on object-bank features. In both cases, we improve over the hand-crafted spatial pooling step showing the importance of its adaptation to the task.

6.1 Introduction

Spatial pooling plays a crucial role in modern object recognition and detection systems. Motivated from biology [Hubel and Wiesel 1962; Fukushima and Miyake 1982; LeCun et al. 1990; Riesenhuber and Poggio 2009] and statistics of locally orderless images [Koenderink and Van Doorn 1999], the spatial pooling approach has been found useful as an intermediate step of many today's computer vision methods ranging from local features based approaches [Lazebnik et al. 2006; Yang et al. 2009] to higher-level semantic representations Li-Jia et al. [2010]. In order to form more robust features under translation or small object deformations, activations of codes and features are pooled over larger areas in a spatial pyramid scheme [Lazebnik et al. 2006; Yang et al. 2009] via a sum or max operator. Unfortunately, this critical decision, namely the spatial division, is most prominently based on hand-crafted layouts and therefore data and task independent.

We propose a flexible parameterization that allows for a richer set of possible pooling regions and show results on classification tasks using two different pipelines [Coates and Ng 2011; Li-Jia et al. 2010]. Moreover, we extend the learnable pooling regions [Malinowski and Fritz 2013b] to the events recognition task with object banks as high level features. The representation is learned jointly with the classifier to support the recognition task. In order to deal with the increased flexibility of the model, we investigate different regularizers and efficient learning schemes. In particular, we propose a smoothness regularizer that yields the strongest performance improvements in our experiments.

6.2 Related Work

There is an increasing interest to push the boundary of learning based approaches towards fully optimized and adaptive architectures where design choices, that would potentially constrain or bias a model, are kept to a minimum. Neural networks have a great tradition of approaching hierarchical learning problems and training intermediate representations [Ranzato et al. 2007; Le et al. 2012]. Along this line, we propose a learnable spatial pooling strategy that can discriminatively shape the pooling regions. In contrast to convolutional neural architectures [Ranzato et al. 2007], our particular architecture has a direct interpretation as a global pooling strategy and therefore subsumes popular spatial pyramids as a special case. Yet we have the freedom to investigate different regularization terms that lead to new pooling strategies when optimized jointly with the classifier.

Recent progress has been made in learning pooling regions in the context of image classification using the Spatial Pyramid Matching (SPM) pipeline [Lazebnik et al. 2006; Yang et al. 2009]. Some researchers [Jia and Huang 2011; Jia et al. 2012; Russakovsky et al. 2012; Sánchez et al. 2012; Feng et al. 2011; Krapac et al. 2011; Koniusz and Mikolajczyk 2011] have further investigated how to liberate the recognition from preconceptions of the hand crafted recognition pipelines. However, these methods still make quite strong assumptions on the solutions that can be achieved. For instance Jia and Huang [2011] optimizes binary pooling strategies that are given by the superposition of rectangular basis functions, and Feng et al. [2011] finds pooling regions by applying a linear discriminant

analysis for individual pooling strategies and training a classifier afterwards. Krapac et al. [2011] and Koniusz and Mikolajczyk [2011] model spatial location of the visual words by fitting Mixture of Gaussians. Russakovsky et al. [2012] and Sánchez et al. [2012] have shown improvement over SPM by pooling the objects and background separately. Although the last two methods are image-dependent they strongly depend on the object localization which is a non-trivial task if bounding boxes are absent during training time. In contrast, our method learns the shape of the pooling region without resorting to the notion of the bounding boxes. However both Russakovsky et al. [2012] and Sánchez et al. [2012] can be combined with our approach as they are complementary. Our method is also complementary to van Gemert [2011] which exploits bias in the photographic style and generalizes SPM to quantize and pool over such attributes as colorfulness, depth of field, viewpoint, lighting, and saliency. In contrast, we learn the pooling regions directly without the use of such additional cues.

6.3 Outline

First, we propose our parameterized pooling operator and show how to jointly optimize the parameters together with the classifier. To cope with the large number of parameters, we investigate regularizers and an efficient learning scheme. We evaluate our method on the CIFAR-10 and show strong improvements in the regime of small dictionaries where our flexible model shows its capability to make best use of the representation by exploring spatial pooling strategies specific to every coordinate of the code. We also show strong classification performance on the CIFAR-100 dataset where our method outperforms, to the best of our knowledge, the state-of-the-art on this dataset in the regime of spatial pyramid architectures. Finally, we also apply our model to higher level events classification tasks that utilize a representation based on object-bank features [Li-Jia et al. 2010].

6.4 Method

In contrast to the methods that use fixed spatial pooling regions in the object classification task [Lazebnik et al. 2006; Yang et al. 2009] our method jointly optimizes both the classifier and the pooling regions. In this way, the learning signal available in the classifier can help shaping the pooling regions in order to arrive at better pooled features.

6.4.1 Parameterized Pooling Operator

The simplest form of the spatial pooling is computing histogram over the whole image. This can be expressed as $\Sigma(\boldsymbol{U}) := \sum_{j=1}^{M} \boldsymbol{u}_j$, where $\boldsymbol{u}_j \in \mathbb{R}^K$ is an encoded patch extracted from the image (out of M such codes) and an index j refers to the spatial location that the code originates from[1]. Another popular pooling scheme that has been proven successful [Yang et al. 2009] is max-pooling: $\mathbb{M}(\boldsymbol{U}) := \max_{j=1}^{M} \boldsymbol{u}_j$. Since the pooling approach looses spatial information of the codes, Lazebnik et al. [2006] proposed to first divide the image into subregions, and afterwards to create pooled features by concatenating histograms computed

[1] That is $j = (x, y)$ where x and y refer to the spatial location of the center of the extracted patch.

over each subregion. There are two problems with such an approach: first, the division is largely arbitrary and in particular independent of the data; second, discretization artifacts occur as spatially nearby codes can belong to two different regions as the 'hard' division is made.

In this paper we address both problems by using a parameterized version of the pooling operator

$$\Theta_{\boldsymbol{w}}(\boldsymbol{U}) := \rho_{j=1}^{M}(\boldsymbol{w}_j \circ \boldsymbol{u}_j) \tag{6.1}$$

where $\boldsymbol{a} \circ \boldsymbol{b}$ is the element-wise multiplication, and ρ is a pooling function. Here, we investigate either sum or max pooling functions and therefore $\rho \in \{\max, \sum\}$. Standard spatial division of the image can be recovered from Equation 6.1 by setting the vectors $\boldsymbol{w}_j$ either to a vector of zeros $\mathbf{0}$, or ones $\mathbf{1}$. For instance, features obtained from dividing the image into 2 subregions using sum pooling can be recovered from Θ by concatenating two vectors: $\sum_{j=1}^{\frac{M}{2}} \mathbf{1} \circ \boldsymbol{u}_j + \sum_{j=\frac{M}{2}+1}^{M} \mathbf{0} \circ \boldsymbol{u}_j$, and $\sum_{j=1}^{\frac{M}{2}} \mathbf{0} \circ \boldsymbol{u}_j + \sum_{j=\frac{M}{2}+1}^{M} \mathbf{1} \circ \boldsymbol{u}_j$, where $\left\{1, ..., \frac{M}{2}\right\}$ and $\left\{\frac{M}{2} + 1, ..., M\right\}$ refer to the first and second half of the image respectively.

In general, let $\mathfrak{F} := \{\Theta_{\boldsymbol{w}}\}_{\boldsymbol{w}}$ be a family of the pooling functions given by Equation 6.1, parameterized by the vector $\boldsymbol{w}$, and let $\boldsymbol{w}^{*,l}$ be the 'best' parameter chosen from the family $\mathfrak{F}$ based on the initial configuration l and a given set of images. First row of Table 6.2 shows four initial configurations that mimic the standard 2-by-2 spatial image division. Every initial configuration can lead to different $\boldsymbol{w}^{*,l}$ as it is shown in Table 6.2. Clearly, the family $\mathfrak{F}$ contains all possible 'soft' and 'hard' spatial divisions of the image, and therefore is their generalization.

6.4.2 Learnable Pooling Regions

In the SPM architectures the pooling weights $\boldsymbol{w}$ are designed by hand, whereas here we aim for joint learning $\boldsymbol{w}$ together with the parameters of the classifier. Intuitively, the classifier during training has access to the classes that the images belong to, and therefore can shape the pooling regions. On the other hand, the method aggregates statistics of the codes over such learned regions and pass them to the classifier allowing to achieve higher accuracy. Such joint training of the classifier and the pooling regions can be done by adapting the backpropagation algorithm [Bishop 1999; LeCun et al. 1998a], and so can be interpreted as a densely connected multilayer perceptron [Collobert and Bengio 2004; Bishop 1999].

Consider a sampling scheme and an encoding method producing M codes each K dimensional. Every coordinate of the code is an input layer for the multilayer perceptron. Then we connect every j-th input unit at the layer k to the l-th pooling unit a_l^k via the relation $w_{lj}^k u_j^k$. Since the receptive field of the pooling unit a_l^k consists of all codes at the layer k, we have $a_l^k := \sum_{j=1}^{M} w_{lj}^k u_j^k$ or $a_l^k := \max_{j=1}^{M} w_{lj}^k u_j^k$, and so in the vector notation

$$\boldsymbol{a}_l := \rho_{j=1}^{M}(\boldsymbol{w}_j^l \circ \boldsymbol{u}_j) = \Theta_{\boldsymbol{w}^l}(\boldsymbol{U}) \tag{6.2}$$

Next, we connect all pooling units with the classifier allowing the information to circulate between the pooling layers and the classifier. We use logistic regression which is connected

to the pooling units via the formula

$$J(\boldsymbol{\Theta}) := -\frac{1}{D}\sum_{i=1}^{D}\sum_{j=1}^{C}\mathbf{1}\{y^{(i)}=j\}\log p(y^{(i)}=j|\boldsymbol{a}^{(i)};\boldsymbol{\Theta}) \tag{6.3}$$

where D denotes the number of all images, C is the number of all classes, $y^{(i)}$ is a label assigned to the i-th input image, and $\boldsymbol{a}^{(i)}$ are responses from the 'stacked' pooling units $[\boldsymbol{a}_l]_l$ for the i-th image[2]. We use the logistic function to represent the probabilities: $p(y = j|\boldsymbol{x};\boldsymbol{\Theta}) := \frac{\exp(\boldsymbol{\theta}_j^T\boldsymbol{x})}{\sum_{l=1}^{C}\exp(\boldsymbol{\theta}_l^T\boldsymbol{x})}$. Since the classifier is connected to the pooling units, our task is to learn jointly the pooling parameters $\boldsymbol{W}$ together with the classifier parameters $\boldsymbol{\Theta}$, where $\boldsymbol{W}$ is the matrix containing all pooling weights. Finally, we use standard gradient descent algorithm that updates the parameters using the following fixed point iteration

$$\boldsymbol{X}^{t+1} := \boldsymbol{X}^t - \gamma\nabla J(\boldsymbol{X}^t) \tag{6.4}$$

where in our case $\boldsymbol{X}$ is a vector consisting of the pooling parameters $\boldsymbol{W}$ and the classifier parameters $\boldsymbol{\Theta}$.

6.4.3 Regularization Terms

In order to improve the generalization, we introduce regularization of our model as we deal with a large number of the parameters. For the classification $\boldsymbol{\Theta}$ and pooling parameters $\boldsymbol{W}$, we employ L_2 regularization terms: $||\boldsymbol{\Theta}||_{l_2}^2$ and $\sum_k ||\boldsymbol{W}^k||_{l_2}^2$. In order to maintain interpretable pooling regions we constraint the solution to the unit cube. This is implemented via projects onto the cube during the optimization. To reduce quantization artifacts of the pooling strategy as well as to ensure smoothness of the output w.r.t. small translations of the image, the model penalizes weights whenever the pooling region is non-smooth. This can be done by measuring the spatial variation $||\nabla_x\boldsymbol{W}^k||_{l_2}^2 + ||\nabla_y\boldsymbol{W}^k||_{l_2}^2$ for every layer k. Therefore our overall optimization objective is

$$\begin{aligned}
&\underset{\boldsymbol{W},\boldsymbol{\Theta}}{\text{minimize}}\, J_{\mathrm{R}}(\boldsymbol{\Theta},\boldsymbol{W}) := \\
&-\frac{1}{D}\sum_{i=1}^{D}\sum_{j=1}^{C}\mathbf{1}\{y^{(i)}=j\}\log p(y^{(i)}=j|\boldsymbol{a}^{(i)};\boldsymbol{\Theta}) \\
&+\frac{\alpha_1}{2}||\boldsymbol{\Theta}||_{l_2}^2 + \frac{\alpha_2}{2}||\boldsymbol{W}||_{l_2}^2 \\
&+\frac{\alpha_3}{2}\left(||\nabla_x\boldsymbol{W}||_{l_2}^2 + ||\nabla_y\boldsymbol{W}||_{l_2}^2\right) \\
&\\
&\text{subject to } \boldsymbol{W}\in[0,1]^{K\times M\times L}
\end{aligned} \tag{6.5}$$

where $\boldsymbol{a}_l$ is the l-th pooling unit described by Equation 6.2, and $||\boldsymbol{W}||_{l_2}$ is the Frobenius norm.

[2]Providing the codes $\boldsymbol{U}^{(i)}$ are collected from the i-th image and $\boldsymbol{a}_l^{(i)} := \Theta_{\boldsymbol{w}^l}(\boldsymbol{U}^{(i)})$ then $\boldsymbol{a}^{(i)} := [\boldsymbol{a}_l^{(i)}]_l$.

6.4.4 Approximation of the Model

The presented approach is demanding to train in the means of the CPU time and memory storage when using high dimensional representations. The number of the pooling parameters to learn grows as $K \times M \times L$, where K is dimensionality of codes, M is the number of patches taken from the image and L is the number of pooling units. Therefore, we propose two approximations to our method making the whole approach scalable to large dictionaries. However, we emphasize that learned pooling regions have very little if any overhead compared to standard spatial division approaches at test time.

The first approximation does a fine-grained spatial partition of the image (3 by 3 pixels), and then pools the codes over such subregions. This operation reduces the number of spatial locations by the factor of the pre-pooling size. The second approximation divides a K dimensional code into $\frac{K}{D}$ batches, each D dimensional. Then we train our model on all such batches in parallel to obtain the pooling weights. Afterwards, we train the classifier on top of the concatenation of the trained, partial models. We also consider a redundant set of such batches in our experiments in order to compensate for potential approximation errors. As opposed to the approximations proposed by Le et al. [2012], our training is fully parallel and doesn't need communication between different batches/machines. In addition, the training of the small models per batch shows on average 5 times faster convergence than the full models.

Implementation details To learn the parameters of the model we use the limited-memory BFGS algorithm[3]. The hyperparameters were selected by 5-fold cross-validation. Our implementation is available at `http://mpii.de/learning-smooth-pooling-regions`.

6.5 Experimental Results

First, we evaluate our method on the CIFAR-10 and CIFAR-100 object recognition datasets [Krizhevsky and Hinton 2010]. Furthermore, we provide insights into the learned pooling strategies as well as investigate transfer between datasets. Second, we show that our method also translates to a high level recognition task of events in a max pooling setting with object bank features [Li-Jia et al. 2010] on the UIUC sports events dataset. [Li-Jia and Fei-Fei 2007]. We start by describing our experimental setup.

Datasets The CIFAR-10 and CIFAR-100 datasets contain 50000 training color images and 10000 test color images from respectively 10 and 100 categories, with 6000 and 600 images per class respectively. All images have the same size: 32×32 pixels, and were sampled from the 80 million tiny images dataset [Torralba et al. 2008]. UIUC sports events [Li-Jia and Fei-Fei 2007] is a dataset containing 8 sports categories such as rowing, badminton, polo, bocce, snowboarding, croquet, sailing, and rock climbing. The number of images varies per class from 137 to 250. We follow Li-Jia et al. [2010] and use 70 images per class for training, and 60 images per class for testing.

[3] implementation by Mark Schmidt: http://www.di.ens.fr/~mschmidt/Software/minFunc.html

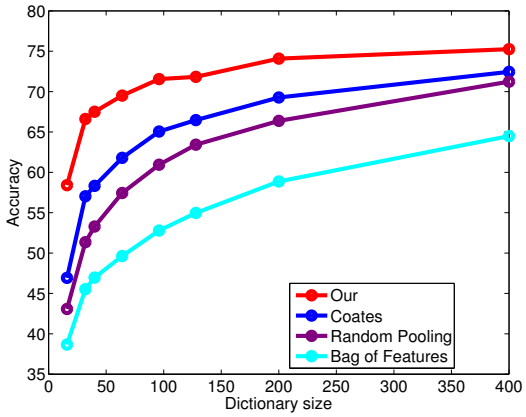
Feature representations In order to insure comparability we follow the evaluation pipeline of Coates and Ng [2011] for the object recognition experiment. We extract normalized and whitened 6×6 patches from images using a dense, equispaced grid with a unit sample spacing. As the next step, we employ the K-means assignment and triangle encoding [Coates and Ng 2011; Coates et al. 2011] to compute codes – a K-dimensional representation of the patch. As we want to be comparable to Coates et al. [2011], who uses a spatial division into 2-by-2 subregions which results in $4 \cdot K$ pooled features, we use 4 pooling units, too. Furthermore, we use a standard division (first row of Table 6.2) as an initialization of our model. In addition to the Coates and Ng [2011] pipeline, we also apply our architecture to max pooling and object banks [Li-Jia et al. 2010]. The latter use object filters [Felzenszwalb et al. 2008] and spatial pyramid matching [Lazebnik et al. 2006; Yang et al. 2009] to build a high-level representation of the image. For both feature representations we use the source code provided by the authors.

Evaluation of our method on small dictionaries Figure 6.1a shows the classification accuracy of our full method against the baseline [Coates and Ng 2011]. Since we train the pooling regions without any approximations in this set of experiments the results are limited to dictionary sizes up to 800. Our method outperforms the approach of Coates by 10% for dictionary size 16 (our method achieves the accuracy 57.07%, whereas the baseline only 46.93%). This improvement is consistent up to the bigger dictionaries although the margin is getting smaller. Our method is about 2.5% and 1.88% better than the baseline for 400 and 800 dictionary elements respectively.

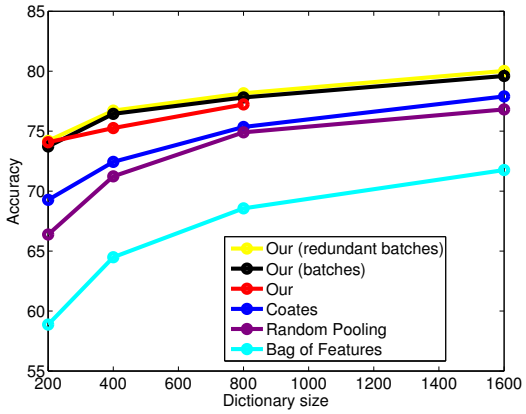
Scaling up to sizable dictionaries In Section 6.4.4 we have discussed how to divide the codes into low dimensional batches and learn the pooling regions on those. In the following experiments we use batches with 40 entries extracted from the original code, as those fit conveniently into the memory of a single, standard machine (about 5 Gbytes for the main data) and can all be trained in parallel.

Besides a reduction in the memory requirements, the batches have shown multiple benefits in practice due to smaller number of parameters. We need less computations per iterations as well as observe faster convergence. Figure 6.1b shows the classification performance for larger dictionaries where we examined the full model [Our], the baseline [Coates], random pooling regions (described in Section 6.5), bag of features, and two possible approximation - the batched model [Our (batches)], and the redundantly batched model [Our (redundant batches)].

Our test results are presented in Table 6.1. We observe little if any drop in accuracy when using our approximation scheme. We attribute this to the better conditioned learning problem of the smaller codes within one batch. With an accuracy for the batched model of 79.6% we outperform the Coates baseline by 1.7%. Interestingly, we gain another small improvement to 80.02% by adding redundant batches which amounts to a total improvement of 2.12% compared to the baseline. Our method performs comparable to the pooling strategy of Jia and Huang [2011] which uses more restrictive assumptions on the pooling regions and employs feature selection algorithm. To the best of our knowledge Goodfellow et al. [2013]



(a) Small dictionaries.



(b) Larger dictionaries.

Figure 6.1: Figure 6.1a shows accuracy of the classification with respect to the number of dictionary elements on smaller dictionaries. Figure 6.1b shows the accuracy of the classification for bigger dictionaries when batches, and the redundant batches were used. Experiments are done on CIFAR-10.

achieves the best results on the CIFAR-10 dataset with an accuracy 90.62% with a method based on convolutional maxout networks architecture and data augmentation – different from global pooling architectures that we investigate in our study.

Random pooling regions Our investigation also includes results using random pooling regions where the weights for the parameterized operator (Equation 6.2) were sampled from normal distribution with mean 0.5 and standard deviation 0.1, that is $\boldsymbol{w}_j^l \sim \mathcal{N}(0.5, 0.1)$ for all l. This notion of the random pooling differs from the Jia et al. [2012] where random selection of rectangles is used. The experiments show that the random pooling regions can compete with the standard spatial pooling (Figures 6.1a and 6.1b) on the CIFAR-10 dataset, and suggest that random projection can still preserve some spatial information. This is especially visible in the regime of bigger dictionaries where the difference is only 1.09%.

Method	Dict. size	Features	Acc.
Jia	1600	6400	80.17%
Coates	1600	6400	77.9%
Our (batches)	1600	6400	79.6%
Our (redundant)	1600	12800	80.02%

Table 6.1: Comparison of our methods against the baseline Coates and Ng [2011] and Jia and Huang [2011] with respect to the dictionary size, number of features and the test accuracy on CIFAR-10.

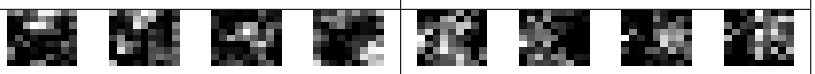
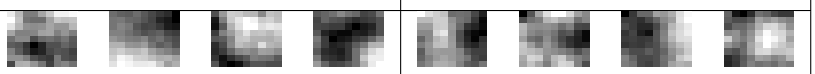
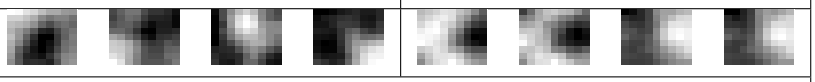
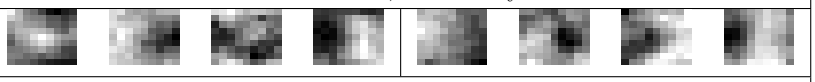
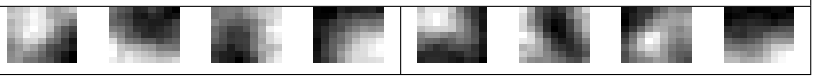
regularization	pooling weights
dataset: CIFAR-10 ; dictionary size: 200	
Coates (no learn.)	
l2	
smooth	
smooth & l2	
dataset: CIFAR-10 ; dictionary size: 1600	
smooth & batches	
dataset: CIFAR-100 ; dictionary size: 1600	
smooth & batches	

Table 6.2: Visualization of different pooling strategies obtained for different regularizations, datasets and dictionary size. Every column shows the regions from two different coordinates of the codes. First row presents the initial configuration also used in standard hand-crafted pooling methods. Brighter regions denote larger weights.

The obtained results indicate that hand-crafted division of the image into subregions is questionable, and call for a learning-based approach.

Investigation of the regularization terms Our model (Equation 6.5) comes with two regularization terms associated with the pooling weights, each imposing different assumptions on the pooling regions. Hence, it is interesting to investigate their role in the classification task by considering all possible subsets of $\{\text{l2}, \text{smooth}\}$, where "l2" and "smooth" refer to $||\boldsymbol{W}||_{l_2}^2$ and $\left(||\nabla_x \boldsymbol{W}||_{l_2}^2 + ||\nabla_y \boldsymbol{W}||_{l_2}^2\right)$ respectively. Table 6.3 shows our results on CIFAR-10. We choose a dictionary size of 200 for these experiments, so that we can evaluate different regularization terms without any approximations. We conclude that the spatial smoothness regularization term is crucial to achieve a good predictive performance of our method whereas the l2-norm term can be left out, and thus also reducing the number of hyper-parameters. Based on the cross-validation results (second column of Table 6.3), we select this setting for

Regularization	CV Acc.	Test Acc.
free	68.48%	69.59%
l2	67.86%	68.39%
smooth	73.36%	73.96%
l2 + smooth	70.42%	70.32%

Table 6.3: We investigate the impact of the regularization terms on the CIFAR-10 dataset with dictionary size equals to 200. Term "free" denotes the objective function without both regularization terms. The cross-validation accuracy and test accuracy are shown.

Method	Dict. size	Features	Acc.
Jia	1600	6400	54.88%
Coates	1600	6400	51.66%
Our (batches)	1600	6400	56.29%

Table 6.4: The classification accuracy on CIFAR-100, where our method is compared against the Coates and Ng [2011] and Jia and Huang [2011].

further experiments.

Experiments on the CIFAR-100 dataset We also investigate how the model performs on more demanding CIFAR-100 dataset with 100 classes. Our model with the spatial smoothness regularization term on the 40 dimensional batches achieves 56.29% accuracy. To our best knowledge, this result constitutes the state-of-the-art performance on this dataset in the regime of SPM architecture, outperforming Jia and Huang [2011] by 1.41%, and the baseline by 4.63%. Non-global pooling schemes like the convolutional max-out networks have recently achieved a performance of up to 61.43% [Goodfellow et al. 2013].

Transfer of the pooling regions between datasets Beyond the standard classification task, we also examine if the learned pooling regions are transferrable between datasets. In this scenario the pooling regions are first trained on the source dataset and then used on the target dataset to train a new classifier. We use dictionary of 1600 with 40-dimensional batches. Our results (Table 6.5) suggest that the learned pooling regions are indeed transferable between both datasets. While we observe a decrease in performance when learning the pooling strategy on the less diverse CIFAR-10 dataset, we do see improvements for learning on the richer CIFAR-100 dataset. We arrive at a test accuracy of 80.35% which is an additional improvement of 0.75% and 0.18% over our best results (batch-based approximation) and Jia and Huang [2011] respectively.

Visualization and analysis of pooling strategies Table 6.2 visualizes different pooling strategies investigated in this paper. The first row shows the widely used rectangular spatial division of the image. The other visualizations correspond to pooling weights discovered by our model using different regularization terms, datasets and dictionary size. The second row shows the results on CIFAR-10 with the "l2" regularization term. The pooling is most

Source	Target	Accuracy
CIFAR-10	CIFAR-100	52.86%
CIFAR-100	CIFAR-10	80.35%

Table 6.5: We train the pooling regions on the 'Source' dataset. Next, we use such regions to train the classifier on the 'Target' dataset where the test accuracy is reported.

	UIUC sports
Object Banks + SPM [Li-Jia et al. 2010]	76.3%
Object Banks + our method	79.4%

Table 6.6: Our approach described in Section 6.4 with max pooling function and object banks.

distinct from the other results, as it learns highly localized weights. This pooling strategy has also performed the worst in our investigation (Table 6.3). The "smooth" pooling performs the best. We see that weights are localized but vary smoothly over the image. The weights expose a bias towards initialization shown in the first row. All methods with the spatial smoothness regularization tend to focus on similar parts of the image, however "l2 & smooth" is more conservative in spreading out the weights. The last two rows show weights trained using our approximation. Visual inspection shows a similar level of localization and smoothness to the regions obtained without approximation. This further supports the use of our division into independent batches.

Results using object banks Lastly, we investigate event recognition on the UIUC Sports database based on object bank features. Li-Jia et al. [2010] proposes a spatial pyramid matching architecture on top of the object bank features – which makes it an application target for our learned pooling regions. Please note that this setting is quite different form the previous task as high level event recognition is addressed and we optimize pooling regions in a max pooling context. In the experiments we use 4 pooling units with max pooling function on top of the response maps from the object bank filters [Li-Jia et al. 2010; Felzenszwalb et al. 2008]. Our results (Table 6.6) show the importance of adaptive approaches also in this high level recognition context. We improve the results from [Li-Jia et al. 2010] that use a hand crafted SPM architecture by 3.1%.

6.6 Conclusion

In this paper we propose a flexible parameterization of global pooling operators which can be trained jointly with the classifier. We study the effect of different regularizers showing the importance of the smoothness. To train the large set of parameters we propose approximations to our model allowing efficient and parallel training without loss of accuracy. Our method outperforms popular hand-crafted pooling-based methods. While our improvements are consistent over the whole range of dictionary sizes, the margin is most impressive for small dictionaries with the improvement up to 10% compared to the baseline [Coates and Ng

2011]. Finally, we apply our method and improve over SPM to high level event recognition using object-banks representation. We believe that our method is a flexible framework to further investigate different pooling strategies and is broadly applicable in spatial pooling architectures.

Chapter 7

A Pooling Approach to Modelling Spatial Relations for Image Retrieval and Annotation

Contents

Over the last two decades we have witnessed strong progress on modeling visual object classes, scenes and attributes that have significantly contributed to automated image understanding. On the other hand, surprisingly little progress has been made on incorporating a spatial representation and reasoning in the inference process. In this chapter, we propose a pooling interpretation of spatial relations that we presented in Chapter 6. Next, we show how it improves image retrieval and annotations tasks involving spatial language. Due to the complexity of the spatial language, we argue for a learning-based approach that acquires a representation of spatial relations by learning parameters of the pooling operator. We show improvements on previous work on two datasets, two different tasks, and two different methods. The first method is shown in greater detail in Appendix A. This architecture creates a recursive network with a topology predicted by a parser. The second method extends the bi-directional image-to-text architecture of Karpathy et al. [2014] to explicitly do spatial reasoning. Finally, we provide additional insights on a new dataset with an explicit focus on spatial relations. This work is a precursor that led us to work on Visual Turing Test (Chapter 8).

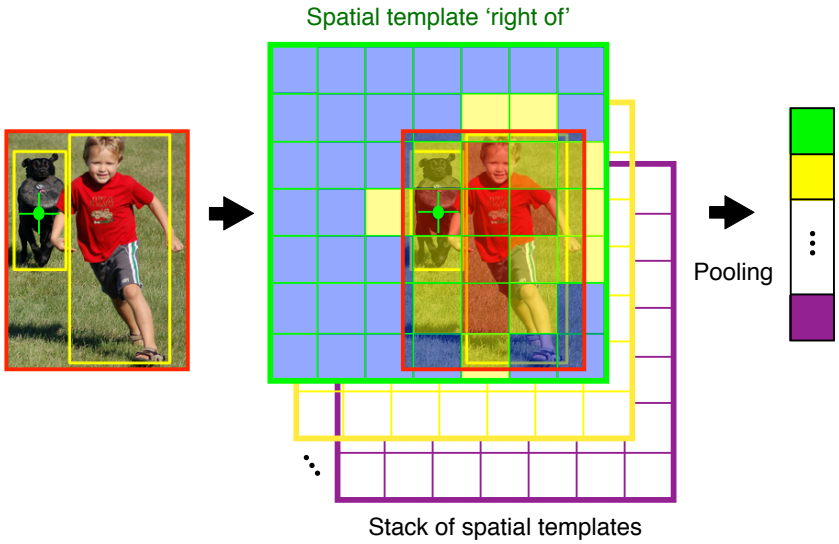


Figure 7.1: We propose a pooling regions interpretation of deictic spatial relations, and show its importance for image retrieval and annotation tasks. We start from a spatial fragment representing a pair of detections: 'boy' and 'dog', and compute spatial representation by projecting the weighted pooling template at the center of the 'dog' detection and pooling the 'boy' localization accordingly.

7.1 Introduction

In a daily life spatial concepts play an important role in human communication. Our comprehension and shared understanding of spatial concepts allow us to make references to specific objects as well as to resolve references made by others. The resolution of such references consists of two aspects, a linguistic part that expresses a relations and the involved concepts and perceptual part that allows us to perceive candidate entities that are involved in the mentioned relations. With spatial relations we can precisely localize object of our interest, ask an another person to act on that object, and expect from the person that first she understands the language of spatial relations and second she has a similar understanding of spatial relations in the environment. As we aim at building machines that "understand" and act upon our intention expressed in natural language, we need to also take care of learning spatial concepts from human data so that both – machine and human – refer to a common apprehension of spatial concepts that are well aligned with each other.

Recent work that has addressed spatial language includes natural language commands for robotics [Tellex et al. 2011; Guadarrama et al. 2013b] and question answering systems about the content of real-world scenes [Malinowski and Fritz 2014a] which relies on hand-crafted approach to spatial representations – often driven by the need for high precision. However, it is also arguable beneficial for problems requiring high recall such as image search [Hodosh et al. 2013; Lan et al. 2012] where coverage on a wide range of spatial concepts becomes important. Yet we are missing techniques to automatically acquire and learn spatial relations to provide the desired coverage.

Apart from building spatial representations in machine perception, there is a long standing

interest from psychologists in understanding how human apprehend spatial concepts [Logan and Sadler 1996; Regier and Carlson 2001]. Mainly based on differences in reference frames, they categorize spatial concepts into *basic*, *deictic* and *intrinsic* relations. Moreover, the psychological studies also offer an interesting model of spatial relations, so called *spatial templates* [Logan and Sadler 1996]. In our work, we are interested in deriving representations of deictic spatial relations and their application to today's image retrieval and annotation methods. These relations express the position of one object with respect to other objects by projecting the observer's frame of reference onto the reference object, and can be modeled with spatial templates. Conceptually, a spatial template is associated with a spatial relation and represents regions of acceptability under the relation. It is centered at the object of reference and computes a goodness of the localization of another object with respect to the referent.

In our work, we exploit that those models of spatial concepts are tightly related to the widely used pooling approaches in computer vision. We show in Section 7.3.1, spatial templates fit into a spatial pooling regions framework [Lazebnik et al. 2006] by fusing ideas of learning pooling operators [Malinowski and Fritz 2013a] with object-centrism [Russakovsky et al. 2012].

Finally, we show that our approach to spatial reasoning readily extends two popular retrieval architectures [Lan et al. 2012; Karpathy et al. 2014] by showing a competitive or even improved results on a two datasets. We also further analyze our model on a new datasets with an explicit focus on spatial relations.

Contributions In this work, we show how spatial pooling regions can be used for spatial representations and reasoning by drawing a link between pooling operators and spatial templates. Next, we show that the spatial templates can be estimated from data if bounding boxes are available and there are spatial sentences of the form (object, spatial relation, object) associated with images. We estimate templates from two sources: our new data with human annotations, and data with automatically generated annotations according to some rules [Lan et al. 2012] and point out differences in the obtained templates. The estimation procedures resembles the experimental scenarios in Logan and Sadler [1996] but results are obtained from real-world images with many different object categories and implicit annotations of spatial arrangement. Finally, we extend two retrieval architecture Lan et al. [2012] and Karpathy et al. [2014] to work with our spatial model. We show how an explicit representation of spatial relations improves performance quantitatively as well as qualitatively by showing the association between language and object on example images.

7.2 Related work

Modeling spatial relations in images Previous work has addressed the problem of image retrieval with structured object queries [Lan et al. 2012] where the authors consider structured queries - a textual input with a binary spatial preposition between two nouns - together with a limited number of different spatial prepositions. Our work goes beyond structured queries and limited spatial vocabulary. For this purpose instead of using a hand-

crafted representation of a set of only few relations ('above', 'below', and 'overlap' like in Lan et al. [2012]), we propose a flexible and learnable representation that is based on spatial templates [Logan and Sadler 1996], and thus can be interpreted as a version of the learnable pooling regions [Malinowski and Fritz 2013a] centered at the reference object.

Image-sentence alignment While there have been successful methods that align sentences with images [Lin et al. 2014a; Kong et al. 2014] the recent research on embedding [Socher et al. 2014; Karpathy et al. 2014; Mao et al. 2014] have opened a door for bi-directional methods that retrieve images based on a textual input, or sentences from a given image. However, in contrast to our work, none of these methods use spatial reasoning to improve the alignment. Karpathy et al. [2014] learns an embedding between textual and visual fragments, while other approaches between an image and a whole sentence.

Spatial pooling regions Spatial pooling has been proven to work well in many recognition tasks [Lazebnik et al. 2006; Yang et al. 2009] and is still a part of many recent approaches [Krizhevsky et al. 2012]. Although the research literature is densely populated with many variations of a spatial pooling regions framework, to the best of our knowledge there is no work that links pooling regions with spatial reasoning on object detections in a scene. In this work, we fill this gap and show a suitable interpretation of the framework. Closely related to our work is an object-centric pooling [Russakovsky et al. 2012] that relies on the object localization methods to distinguish between a foreground and background and next pool over both regions separately. Although, our method is also based on the localization of different objects, we spatially relate every pair of detections in the image to reason about their spatial arrangement.

Grounding spatial relations Although research on grounding of spatial language has a long standing tradition, previous methods mostly focus on rule-based spatial representation [Moratz and Tenbrink 2006; Kruijff et al. 2007] or more recently on a set of hand-crafted spatial features with learnt weights [Tellex et al. 2010; Golland et al. 2010; Lan et al. 2012; Guadarrama et al. 2013b]. Although the latter approaches show improvements they still rely on designing the right set of features and their generalization and scalability to many spatial relations have not been proven yet. Lan et al. [2012] uses only 2 spatial prepositions, while Golland et al. [2010] and Guadarrama et al. [2013b] concentrate on 11.

In our work, we propose a simple and uniform learning-based approach to spatial representation, and validate the proposed approach on different image-retrieval tasks with many spatial prepositions.

7.3 Method

We are proposing a representation for spatial relations and how it can be applied to image retrieval and annotation. Motivated by the work on spatial templates [Logan and Sadler 1996], we establish a connection between the popular pooling representations and the spatial templates.

First, we present our spatial model and describe how it is parameterized. Then, we present an application of our approach to image retrieval setting [Lan et al. 2012] with a restricted query language and where ground truth bounding boxes of different objects are available. We proceed by showing how our spatial model can be incorporated into a fragment embeddings framework [Karpathy et al. 2014]. Here, annotated bounding boxes are unavailable and the query language is unrestricted.

In Section 7.3.1, we discuss a novel extension of a spatial pooling approach [Lazebnik et al. 2006] to support spatial arrangement between detections. In the following sections we show different instances of our model. In Section 7.3.2, we discuss an application of the spatial templates where bounding boxes of different objects are known during the training and the query language has a restricted structure, while Section 7.3.3 shows how to extend the deep fragment embeddings [Karpathy et al. 2014] to work with spatial templates in unrestrictive setting without ground truth bounding boxes.

7.3.1 Modeling spatial representations by spatial pooling

Spatial basis Spatial pooling framework [Lazebnik et al. 2006] can be interpreted in terms of spatial basis

$$\Theta^k = \sum_{j=1}^{M} \boldsymbol{w}_j^k \circ \boldsymbol{u}_j \tag{7.1}$$

where $\boldsymbol{u}_j$ is an image feature located at position $j = (x, y)$ in the image, $\circ$ is a piece-wise multiplication, and k refers to the k-th spatial pooling template. Hence, the standard spatial pooling with division into 2-by-1 subregions can be phrased in this representation as $\Theta^1 = \sum_{j=1}^{\frac{M}{2}} \mathbf{1} \circ \boldsymbol{u}_j + \sum_{j=\frac{M}{2}+1}^{M} \mathbf{0} \circ \boldsymbol{u}_j$ and $\Theta^2 = \sum_{j=1}^{\frac{M}{2}} \mathbf{0} \circ \boldsymbol{u}_j + \sum_{j=\frac{M}{2}+1}^{M} \mathbf{1} \circ \boldsymbol{u}_j$, where $\left\{1, ..., \frac{M}{2}\right\}$ and $\left\{\frac{M}{2} + 1, ..., M\right\}$ refer to the first and second half of the image respectively. Using such representation, the pooling operator can be included in a learning-based framework where the pooling weights $\{\boldsymbol{w}_j^k\}_{j,k}$ are jointly optimized together with a classifier [Malinowski and Fritz 2013a]. Although, originally the logistic regression is used, the whole method is agnostic to the choice of a classifier and can be easily integrated with other objective functions with an additional hyper-parameter defining the size of the receptive field (or equivalently the discretization level) and the number of the pooling templates Θ^k.

The pooling interpretation of spatial relations In psychology, Logan and Sadler [1996] have proposed a theory of the spatial relations apprehension by estimating a fit of a spatial template. The template is centered at the reference object and models the relative locations of other objects in the environment. Although the theory has existed for a long time in the psychological community, there is little work that includes similar concepts in modern computer vision architecture for a spatial reasoning. The theory identifies spatial templates with different spatial prepositions and represent those as score maps centered at the object of reference. The support of such score map covers the whole environment and it 'softly' computes a spatial fit of a related object to the reference object under the relation by taking the score at the object's position. For instance, all the objects at the right position of the reference object gets a high score under the 'right template' and a low score under the

'left template'. Most strikingly, such templates can be interpreted in terms of the pooling regions with an image as the environment.

Consider a pair of detections representing 'dog' and 'boy' together with a statement 'A boy on the right side of a dog' as shown in Figure 7.1. Let x, y be the center of the 'dog' bounding box. Now, we place the center of the weighted spatial pooling regions $\Theta^{\text{right of}}$ at the position x, y and pool over NxM different subregions according to the weights. This produces a feature that characterizes the fit of the localization of 'boy' according to the spatial template 'right of'. Here, N and M characterize the discretization level. Accordingly, our representation of spatial relations is computed as follows:

$$\Theta^{\text{rel}}(i, \boldsymbol{u}^{(d)}) = \sum_{j=1}^{M} w_j^{\text{rel}} \circ u^{(d)}_{i-\frac{M}{2}+j} \tag{7.2}$$

where i is the position of the reference object, $\boldsymbol{u}^{(d)}$ is a score map representing the localization of the related object d (e.g. a detector score map, or introduced in Section 7.3.3 a Dirac image), with value 0 for positions outside of the image. In contrast to Equation 7.1, w_j^{rel} and $u_j^{(d)}$ are scalars. Latter represents the localization score map at position $j = (x, y)$.

In this work, we investigate two special cases of the more general spatial framework in the context of image retrieval. First, in Section 7.3.2 we take advantage of the ground truth bounding boxes and initialize the pooling weights with the estimated spatial spatial templates (Table 7.2). In this scenario, we use queries with a limited structure and vocabulary. Second, in Section 7.3.3 we consider a challenging scenario with a complex natural language queries and where ground truth bounding boxes are missing.

7.3.2 Estimating spatial templates

We consider a scenario with a restricted query language of the following form (noun, spatial preposition, noun) together with a limited vocabulary without inflection - for instance ('airplane', 'in front of', 'building'). Moreover, let assume the annotated bounding boxes are available during the training with the object categories from the same vocabulary. Thanks to those restrictions, and in contrast to Section 7.3.3, we can first estimate the spatial templates from data, and next initialize the pooling weights $\{w_j^k\}_{j,k}$ with the estimations.

To exemplify the estimation procedure, consider a spatial preposition 'above' and take all the images that are annotated with a sentence containing 'above', for instance ('picture','above','bed'). Next, we center a spatial template representing 'above' at the center of 'bed' bounding box and copy the content of the 'picture' bounding box. Afterwards, we proceed to the subsequent image with 'above' annotation and repeat the 'copying' procedure while storing the already copied contents. To obtain smooth spatial templates, we create the localization score map by filling the whole 'picture' bounding box with ones and take it as its content. Finally, we use such derived spatial template as the initialization of $\{w_j^{\text{above}}\}_j$. Table 7.2 shows the estimated spatial templates for spatial relations that we use in our experiments. Since the initialization already acts as a strong regularization, unlike in Section 7.3.3, we do not need to resort to discretization of the image space into large subregions - in other words we consider one pixel sized receptive fields. Note that our

estimation is still based solely on the descriptions of the image and does not require directly annotating spatial relations.

In Section 7.4 we visually inspect the estimated templates. Interestingly, the estimation procedure and our visualizations resemble the experimental scenarios in Logan and Sadler [1996] where templates are estimated from the points drawn by humans on a frame with respect to a given spatial preposition. Our case is however different in that we collate results based on real world images with many object categories and implicit spatial arrangement. That is, for every sentence of the form (object, spatial relation, object), participants of the experiment only annotated which images satisfy the sentence.

Next section shows how to include a spatial model into the state-of-the-art method on a retrieval task with missing ground truth bounding boxes and unconstrained language.

7.3.3 Deep fragment embeddings with spatial reasoning

Deep fragment embeddings The main goal of Karpathy et al. [2014] is to retrieve relevant images based on a sentence query, and conversely. The model learns a bi-directional embedding on a set of unconstrained images and corresponding sentences. As opposed to previous work on embedding, it finds a mapping between visual fragments represented as the image-induced activations of the bottleneck layer of the most certain detections [Girshick et al. 2014], and textual fragments that are represented as triplets of the form $(R, \boldsymbol{t}_1, \boldsymbol{t}_2)$, where $\boldsymbol{t}_1$ and $\boldsymbol{t}_2$ are 1-of-k word encodings under a binary dependency relation R [De Marneffe et al. 2006]. Moreover, the framework does not require any annotated associations between the textual and visual fragments nor even annotated bounding boxes. Instead, it incorporates a MIL [Chen et al. 2006] procedure into the learning process. The objective function consists of two parts: global ranking objective that learns the image-sentence similarities that are consistent with the ground truth annotations, and fragment alignment objective that is based on the intuition that for a given textual fragment at least one of the bounding boxes in the corresponding image should have a high score with this fragment. The learning process optimizes a linear combination of both objectives and aims at finding a good inner-product based similarity between the fragments. For a detailed exposition of the objective function, we refer the reader to Karpathy et al. [2014].

We use both, textual $\boldsymbol{s} = f\left(W_R \left[W_e \boldsymbol{t}_1; W_e \boldsymbol{t}_2\right] + b_R\right)$, and visual fragments $\boldsymbol{v} = W_m \left[\text{CNN}(I_b)\right]$ in our work. Here, W_e is a fixed $400,000 \times 200$ matrix that encodes a 1-of-k vector into a 200-dimensional distributed representation [Huang et al. 2012], f is RELU activation function [Glorot et al. 2011], and $\text{CNN}(I_b)$ is a 4096 dimensional activations of the bottleneck layer induced from the image fragment I_b. The fragment embedding weights W_R, W_m, and b_R are learnt jointly using the aforementioned objective function so that the score $\boldsymbol{v}^T \boldsymbol{s}$ is high for the fragments that match well, low otherwise.

Spatial extension In addition to the visual and textual fragments, we introduce spatial fragments that are based on the pooling interpretation of spatial relations. Let $\Theta^k(O_j, \cdot)$ be a weighted spatial division that represents k-th spatial concept centered at the position of j-th detection. Here, $O_j = (x_j, y_j)$ represents the center of the j-th bounding box. We can formally cast such representation into the spatial pooling framework as follows. Let $\boldsymbol{u}^d$

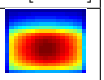
Lan et al. [2012]	
	
above	below

Table 7.1: Visualization of estimated spatial filters. A set of relations from Lan et al. [2012].

Extended set of relations										
										
above	across from	behind	below	in	in front	inside	left	right	on	under

Table 7.2: Visualization of estimated spatial filters. Extended set of relations.

be a Dirac image associated with detection d. It is $u^d_{(x,y)} = 1$ if (x, y) is the center of the bounding box d and $u^d_{(x,y)} = 0$ at other positions. For every pair of detections, we consider the reference detection d_1 and build a Dirac image $\boldsymbol{u}^{d_2}$ of the related detection. Next, we place the spatial template k at O_{d_1} - the center of the reference detection - and pool over the Dirac image $\boldsymbol{u}^{d_2}$, producing a spatial fragment $\Theta^k(O_{d_1}, \boldsymbol{u}^{(d_2)})$. We repeat such procedure for every pair of detections, with the 1st and 2nd elements of the pair as the reference and related detections, finally producing a D^2 such spatial fragments for every spatial concept, where D is the number of detections.

Such representation can be transformed into the matrix-vector multiplication framework, which is consistent with Karpathy et al. [2014], by pulling out the weights and a discretization of the image space: $\boldsymbol{p} = W_s g(\boldsymbol{u}^d)$. Here, $\boldsymbol{u}^d$ is the Dirac image of a detection d, $g(\boldsymbol{u})$ takes a Dirac image $\boldsymbol{u}$, discretize it into N-by-M subregions, and subsequently vectorize it. The matrix W_s is a mapping from NM dimensional vector space into a K dimensional space of spatial concepts. Note that, although this space can directly correspond to K different prepositions, it can also be treated more abstractly with K chosen based on a validation set.

Analogously, we define spatio-textual fragments

$$\boldsymbol{z} = f\left(W_z \left[W_e \boldsymbol{t}_1; W_e \boldsymbol{t}_2\right] + b_z\right) \tag{7.3}$$

where W_z maps from the 400 dimensional representation of both words into a K dimensional space of spatial concept. Finally, we use the same objective function to train the weights so that $\boldsymbol{p}^T \boldsymbol{z}$ give a high score for the matching spatial fragments and a low score otherwise.

7.4 Experiments

We conduct experiments on several datasets. First, two retrieval datasets use a constrained query language that allow us to use annotated bounding boxes during the training. Here, we estimate spatial templates as described in Section 7.3.2. Both datasets augment the SUN09 image dataset with queries. The first dataset is introduced by Lan et al. [2012] and uses automatically generated queries, while the second dataset is our extension of Lan et al. [2012] with a human annotated queries and a wider range of spatial relations. Note that the

difference between both annotation procedures is substantial, as in our dataset we deal with human notion of spatial concepts that are inherently ambiguous. In addition to the queries, both datasets include annotations which images are relevant to a given query. Again, our proposed annotations are based on human judgement. The last and the most challenging dataset, Pascal1k [Rashtchian et al. 2010], is a collection of images with associated natural language sentences. Although it does not contain the relevance annotations, it can still be used for a retrieval task [Socher et al. 2014; Karpathy et al. 2014].

7.4.1 Dataset

Images All our experiments are based on real-world images. The SUN09 dataset [Choi et al. 2010] consists of 12,000 annotated images with more than 200 object categories. We use 4367 images for training and 4317 images for testing - the same split as in Choi et al. [2010] and Lan et al. [2012]. The second dataset consists of 1000 PASCAL images [Everingham et al. 2008; Rashtchian et al. 2010]. Here, we follow Karpathy et al. [2014] and use 800 images for training, 100 for validation, and 100 for test.

Evaluation measures To be consistent and comparable with Lan et al. [2012] we use Mean Average Precision (mAP) across all queries to measure the performance of different methods on our first two datasets. This measure favors the retrievals with high precisions. Similarly, for the sake of consistency with Karpathy et al. [2014], we use Recall@k (R@k) and Mean Rank (mean r) performance measures [Hodosh et al. 2013]. Recall@k computes the fraction of times the correct result is found among the top k retrievals. This measure favors high recall retrievals and is motivated by the search engines where it is more important to retrieve correct retrievals among top k results.

Structured queries Structured queries are introduced in Lan et al. [2012], but were not formally defined. Here, we formalize the notion of structured queries. We say that a query q is structured if it has the form: $q := q_1 \wedge q_2 \wedge ... \wedge q_n$, where q_i denotes either a noun or a triplet (noun, preposition, noun).

Our dataset of structured queries with richer and human-based spatial language We use the structured queries from Lan et al. [2012] of the form (noun, spatial preposition, noun) with spatial prepositions such as 'above' and 'below', and extend such set to have queries with more spatial prepositions: 'left of', 'right of', 'in front of', 'behind', 'inside of', 'on', 'under', 'across from' and 'in'. We collect annotations by first asking in-house annotators to describe randomly selected images from the SUN09 dataset. Only tuples of the form ('noun', 'spatial preposition', 'noun') are permitted. In the second pass we curate this dataset and arrive at 53 structured queries. Finally, the annotators annotate a binary relevance of each image according to every query. Since the latter requires a lot of human effort we have automatized the process by showing only images containing all objects described in a query. In this process, we have collected about 450,000 relevance annotations and 53 structured queries. In both passes, we instruct the annotators to take an observer's frame of reference. Although our dataset uses a more restrictive query language than Rashtchian et al. [2010],

it is still challenging due to the use of human notion of spatial relations and high variations of object appearance in real-world images. Although, ideally we would annotate also all spatial relations in every image, this process turns out to be too expensive as it scales up quadratically wrt. the number of objects in the scene per relation. Therefore, we decide on a more scalable approach where only descriptions of the relations are given.

Compared with Lan et al. [2012], our dataset consists of more spatial prepositions. In additions, our annotations are generated by human annotators while the previous dataset uses a hand-crafted spatial model that is used to generate image descriptions as well as in the inference.

Compared with Rashtchian et al. [2010], our dataset provides a more reliable comparison with ground truth for the image retrieval task due to our relevance annotations. In addition, instead of focusing on the all aspects of the language, it is mostly about spatial relations.

7.4.2 Evaluation

We investigate several experimental scenarios. First, we compare our method against previous work on the structured queries [Lan et al. 2012], where we show that with learnt spatial templates we can achieve comparable results to hand-crafted representations of spatial relations, but under much weaker assumptions. Second, we also establish a baseline on our new dataset with human-based spatial relations and show that our method can learn an extended set of spatial concepts. Third, we show the benefits of using spatial relation during the inference on a complex task with unconstrained natural language queries and real-world images without exploiting ground truth bounding boxes [Rashtchian et al. 2010; Karpathy et al. 2014]. Fourth, we visually investigate the estimated templates, and show improvement in alignment between language fragments and images.

Comparison to previous work on structured queries In order to establish a comparison to previous work on structured queries, we run experiments on the structured queries from Lan et al. [2012] and compare to their approach in Table 7.3. This dataset consists of 862 (463 for training and 399 for testing) queries of the form (noun, preposition, noun) with 111 nouns. Their experiment contains only two different spatial relations: 'above' and 'below'. In this dataset, the spatial relations are automatically extracted by a hand-crafted formula on the (x, y) coordinates of bounding boxes and serve as exact definitions of the spatial relations. This spatial model is also used by the system of Lan et al. [2012] during the inference. In contrast, we assume that the procedure of generating queries is unknown to our system and we aim at obtaining good representations of the spatial relations only from data. The model of Lan et al. [2012] implements a structured SVM approach and models both the spatial relationship between objects in the query and co-occurrence between non-query and query objects via the compatibility function:

$$\sum_{i \in V_q} \alpha_i^T f(I(l_i)) + \sum_{i \in V_q} \sum_{j \in \mathcal{X} \setminus V_q} \gamma_{ij}^T f(I(l_j)) + \sum_{i,j,k \in E_{\mathcal{Q}}} \beta_{ijk} d_{\mathcal{Q}}(l_i, l_j, k) \tag{7.4}$$

Structured queries	
Method	mAP
Part based detector [Felzenszwalb et al. 2010]	7.76%
MARR [Siddiquie et al. 2011]	10.01%
Structure model [Lan et al. 2012]	11.16%
Our model	11.12%
Extended dataset of human queries	
Our model	7.90%

Table 7.3: Performance of our model that uses estimated spatial templates to other baseline approaches. Note that Structure model uses the same rules to generate questions with spatial prepositions and during the inference.

Here, α, γ and β are weights learnt by the classifier, V_q is a set of all objects (nouns) in the query, $\mathcal{X}$ is a set of all objects available during training, $f(I(l_i))$ is a HOG descriptor extracted [Felzenszwalb et al. 2010] at location l_i, $E_{\mathcal{Q}}$ denotes a set of object pairs and their spatial relations present in the query $\mathcal{Q}$, and $d_{\mathcal{Q}}(I_i, I_j, R_k)$ is used spatial model between detections I_i and I_j under the spatial relation R_k. The last term is equal to 1 if detections l_i and l_j are consistent under the spatial relation k, and is equal to 0 otherwise. The consistency is determined via the same set of rules that are used to create queries. This method achieves a performance of 11.16% mAP without global features on queries of the type (noun, spatial preposition, noun). Moreover, we also report the results of two more baselines (special cases of Equation 7.4): Part based detector where the sum of maximum response scores from each object detector is used as a score and the MARR model [Lan et al. 2012]. The latter uses object detections as the features for the classifier and models co-occurrence between the detections (the second term in Equation 7.4), but without a spatial model.

Since we are mostly interested in learning spatial relations, we implement the same compatibility function (Equation 7.4) but with our spatial component $\Theta^k(O_2, \cdot)$ that represents a spatial filter representing preposition k and centered at the localization of the detection with a category pointed by a query (object 1, preposition, object 2). This matching between the category names and queries is possible since both use the same vocabulary for the objects. For the same reason, we use 'preposition' to index different spatial templates, hence K is equal to the size of spatial vocabulary. Note that here, we compute a spatial relationship between a pair of detections with categories extracted from the query. As Table 7.3 shows, our approach achieves comparable results at 11.12% to the state-of-the-art despite the fact that we did not assume knowledge on the underlying representation of the spatial relations that the data was generated with. The first two rightmost entries in Table 7.1 and Table 7.2 show the templates that we have estimated from data to capture a notion of the spatial relations.

Extended set of spatial relations with queries annotated by humans We extend our analysis to our new dataset that contains an extended set of spatial relations that are – in contrast to the previous dataset – collected from human annotations. Since the

exact human notion of spatial concepts is unknown, it has to be acquired from data. The second part of Table 7.3 (Extended dataset of human queries) shows the performance of our approach, which achieves 7.90% mAP, on our collected data with human queries. Note a drop in performance compared to the previous experiments as this is a more challenging setting.

Visualization of spatial templates To gain more insights about the spatial concepts apprehension, we visualize the estimated templates. The eleven entries in Table 7.2 show the spatial templates estimated on our new dataset. They follow our intuitions about the spatial layout (e.g. 'in' and 'inside' templates are much more focused than other spatial templates). More importantly, our visualization suggests that human apprehension of 'above' and 'below' relations clearly differ from the procedure used to generate queries in Lan et al. [2012] and presented in Table 7.1, both are more focused in our case. Interestingly, even if 'below' and 'under' are synonyms, the corresponding templates are not exactly the same. This suggests a slightly different human apprehension of both concepts. Also, pairs 'left'/'right' and 'above'/'below' are not entirely symmetrical. Although, some concepts such as 'in front' or 'behind' are rather three dimensional, it is still interesting to see how humans perceive them in a plain image.

Analysis of retrieved images We show the retrieved images by our architecture given an example query ('plane', 'in front of', 'building'). Figure 7.2 shows the images together with their corresponding ranks. Further analysis revealed that most mistakes come from failure modes of the object detectors that our and Lan et al. [2012]'s methods are based on. Although there are stronger object detectors [Girshick et al. 2014] than part based models [Felzenszwalb et al. 2010], we decide to keep the latter for the sake of consistency with Lan et al. [2012] and since our work is mainly concerned about spatial concepts.

Experiments on Pascal1k with unconstrained queries Our estimates of the spatial templates from the previous sections rely on the restricted language in form of structured queries and annotated bounding boxes. We now turn to the Pascal1k dataset that features natural language sentences and therefore requires us to deal with implicit supervision for learning representations of spatial relations. We improve over Deep Fragment Embeddings [Karpathy et al. 2014][1] to include our spatial model as discussed in Section 7.3.3. For our method, we choose the dimension of a space of spatial concepts (Section 7.3.3) to be 4, and a spatial representation of 20 pooling regions (precisely the 2-by-2 + 4-by-4 scheme) based on the validation set. Here, we treat a space of spatial concepts more abstractly and we do not associate the prepositions with the indices to spatial templates. Our spatial fragments are pairs of detections, and spatio-textual fragments are arbitrary triplets (R, $\boldsymbol{t_1}$, $\boldsymbol{t_2}$) from the dependency parser. We find it more effective to start the training with the only original model and next proceed to a joint training with our spatial extension. Following Karpathy et al. [2014] we also compare our method against other embedding

[1] We downloaded the source code from http://cs.stanford.edu/people/karpathy/defrag/code.zip. Our performance numbers are on average slightly better then the reported ones in Karpathy et al. [2014], as the code has been improved after the publication.

Figure 7.2: Top ranked retrieved images from the query 'An airplane in front of a building' (SUN09 image dataset and our set of human queries). We see a high recall achieved by our method and two clear mistakes - Rank 7 and Rank 15.

models on this dataset. Table 7.4 shows that our model improves over Deep Fragment Embeddings and consistently outperforms other methods on both tasks: image retrieval and image annotation (here the method retrieves sentences based on the image). Adding our spatial model to Deep Fragment Embeddings improves R@10 by 1.4 and 2.0 units on both tasks respectively. We have also implemented spatial model based on the distance and containment features [Golland et al. 2010] but we didn't achieve satisfactory results - the model barely outperforms Deep Fragment Embeddings. Table 7.4 proves the point that the state-of-the-art retrieval architectures benefit from a spatial model that we propose.

Improved and interpretable alignment Given a set of detections representing visual fragments and two words under a dependency relation representing textual fragments, Deep Fragment Embeddings learns a binding so that the dot product between the matching fragments is high. Hence, for a textual fragment (dependency relation, word 1, word 2), we compute the scores between every detection and the textual fragment, and visualize top 4 scoring bindings. As we argue in this work, the notion of fragments can naturally be generalized to pairs of detections that are in a spatial relation. This is particularly attractive because of the symmetry to textual fragments that always take two words under some relation dependency into account. Figure 7.3 shows how alignment improves over the original non-spatial model.

As an example the fragment ('num', 'gentleman', 'two'), which comes from a sentence 'Two gentleman talking in front of propeller plane', aligns well with a spatial fragment representing human detections. Another interesting example includes the fragment ('with', 'jet', 'gear') with the second top fragment that relates the plane's cockpit with its gears (the top scoring one relates two gears together). Such interpretability is often missing in the output of the original model (second and fourth rows of Figure 7.3).

Pascal1k				
Image Retrieval				
Method	R@1	R@5	R@10	Mean r
Random Ranking [Karpathy et al. 2014]	1.6	5.2	10.6	50.0
Socher et al. [Socher et al. 2014]	16.4	46.6	65.6	12.5
kCCA [Socher et al. 2014]	16.4	41.4	58.9	15.9
DeViSE [Frome et al. 2013]	21.6	54.6	72.4	9.5
SDT-RNN [Socher et al. 2014]	25.4	65.2	84.4	7.0
Deep Fragment [Karpathy et al. 2014]	25.0	**69.4**	83.8	6.9
Our model	**29.0**	68.6	**85.2**	**6.7**
Image Annotation				
Method	R@1	R@5	R@10	Mean r
Random Ranking [Karpathy et al. 2014]	4.0	9.0	12.0	71.0
Socher et al. [Socher et al. 2014]	23.0	45.0	63.0	16.9
kCCA [Socher et al. 2014]	21.0	47.0	61.0	18.0
DeViSE [Frome et al. 2013]	17.0	57.0	68.0	11.9
SDT-RNN [Socher et al. 2014]	25.0	56.0	70.0	13.4
Deep Fragment [Karpathy et al. 2014]	37.0	69.0	84.0	10.4
Our model	**38.0**	**70.0**	**86.0**	**10.3**

Table 7.4: Performance of our model that uses a learnable spatial pooling framework to learn the spatial templates. Our method is built on top of Deep Fragments [Karpathy et al. 2014]. R@k is Recall@K (high is good), Mean r is the mean rank (low is good).

7.5 Summary

We address the problem of missing spatial relations in modern retrieval architectures. Although the research on spatial concepts has a long tradition, it mostly concerns robotics. Even then, previous works use either rule-based approaches or a hand-crafted set of features. In contrast, our work links spatial models with spatial pooling regions framework and offer a simple and uniform framework for spatial reasoning. Next, we conduct several experiments where we show that a competitive pooling-based spatial model can be learnt solely from data. Our analysis on newly collected data shows that automatically generated queries from the previous work have different distribution of spatial concepts than the real data. Moreover, our visualization of alignments suggests that spatial model improves bindings between fragments. Finally, we hope that our results together with our data of spatial queries will foster further research on spatial concepts. For this purpose we will make our dataset publicly available. In particular, we are excited to study other spatial categories and higher order spatial terms.

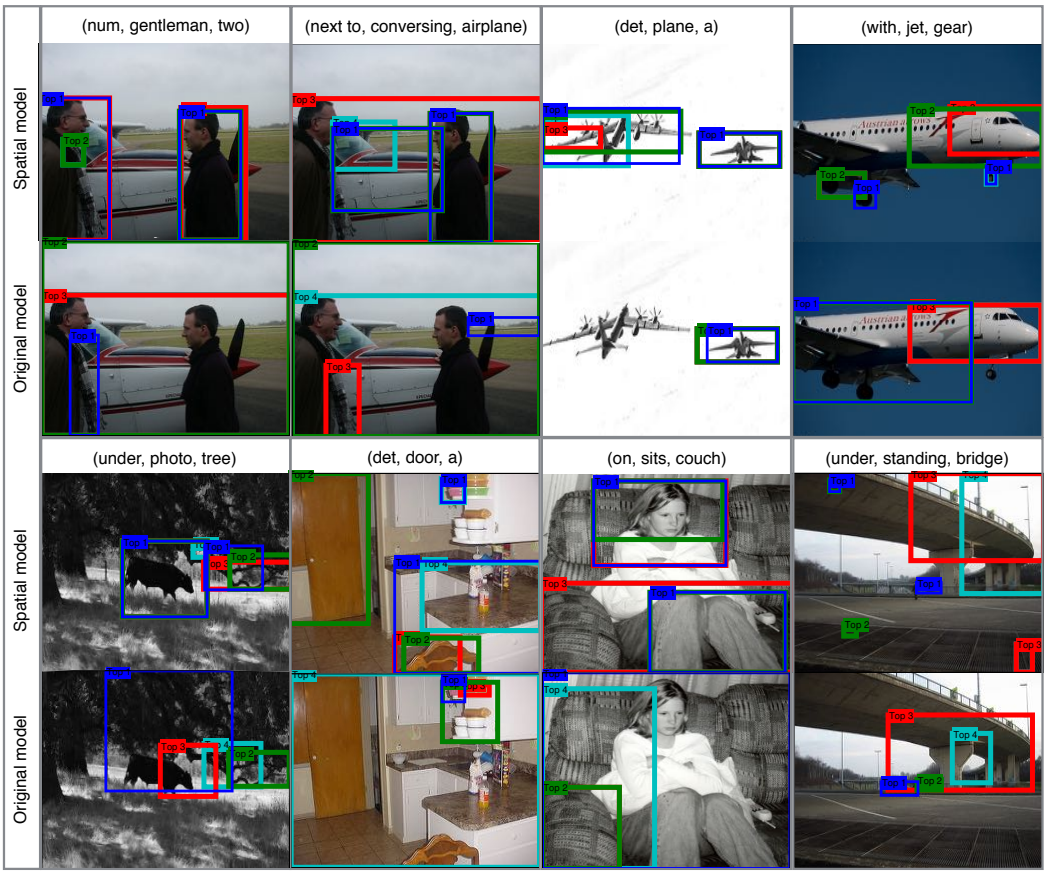


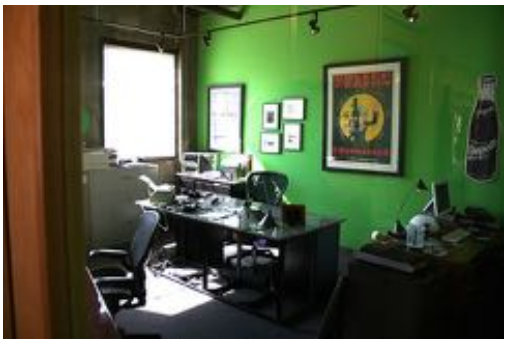
Figure 7.3: Top 4 best bindings between a textual fragment and all detections. Every column represents different textual fragments. The first and third rows show a spatial embedding. The second and fourth rows show an original embedding [Karpathy et al. 2014]. Colors encode scores of fragments associations. Starting from the top scoring: blue, green, red, and cyan. If two fragments overlap, we only show the top scoring one. Since spatial fragments represent pairs of detections, we use the same color encoding for the same pair. Best viewed in color.

7.6 Visual inspection

In this section, we provide further visualizations of the experiments where two models are compared: the (non-spatial) Deep Fragment Embeddings and its spatial extension. The shown results are conducted on the Pascal1k dataset. We show the top 10 ranked retrieved images and top 5 ranked annotations produced by both models. For the sake of uniform visualization, we transformed all images to have equal width and height. Correct retrievals or annotations are shown in red. The qualitative and quantitative results consistently show the benefits of using our spatial extension, where the model learns a spatial arrangement between pairs of detections.

Spatial

1. The corner of a cluttered room with a television and two full book shelves
2. An open bottle of dark ale beer
3. A bottle of beer with the cap taken off
4. A green painted office with a buster eaten poster and a soda bottle on the wall
5. A woman sits with her head down at a table that has alcohol beverages and accessories on it



Non-spatial

1. An open bottle of dark ale beer
2. The corner of a cluttered room with a television and two full book shelves
3. A bottle of beer with the cap taken off
4. A small kitchen with items stacked on the shelves and on the counter
5. Opened bottle of beer

Figure 7.4: Textual retrievals for a given image.

Spatial

1. Several people on bicycles riding over bridge
2. A train is railing between a dead end street and a stand of evergreens
3. A red trolley bus passing by on the opposite side of a city street
4. There is a man riding on the back of a three wheeled bicycle in traffic
5. Three bicyclists crossing a bridge in a city

Non-spatial

1. A train is railing between a dead end street and a stand of evergreens
2. Two bicycle riders about to cross a bridge alongside a rail track
3. A red trolley bus passing by on the opposite side of a city street
4. A school bus is driving uphill on a rural road
5. Big Ben clock in London with red double decker bus driving by

Figure 7.5: Textual retrievals for a given image.

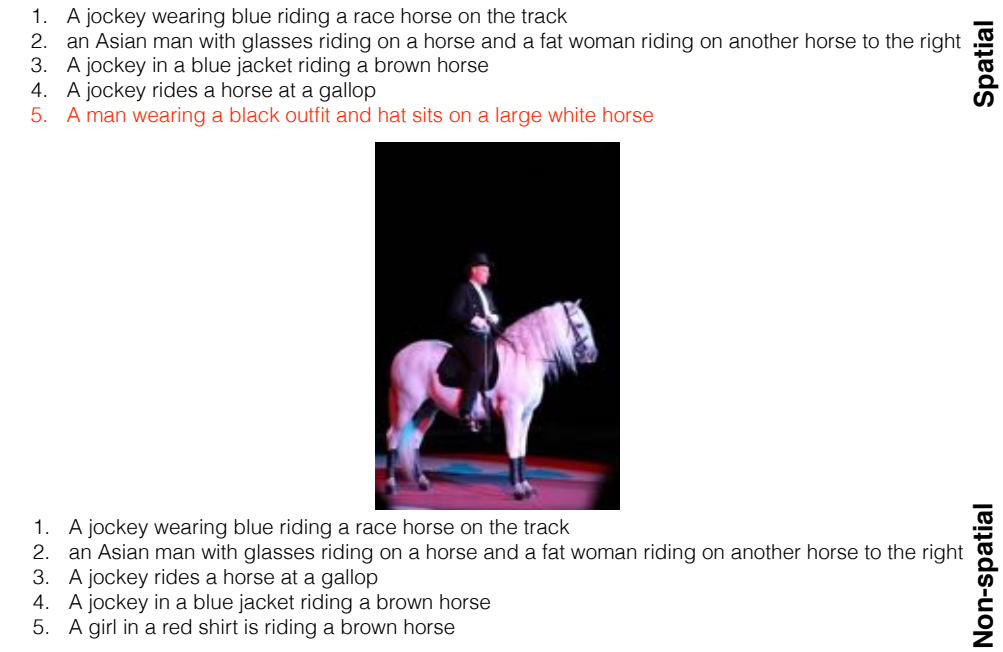


Figure 7.6: Textual retrievals for a given image.

Figure 7.7: Image retrievals for a given query.

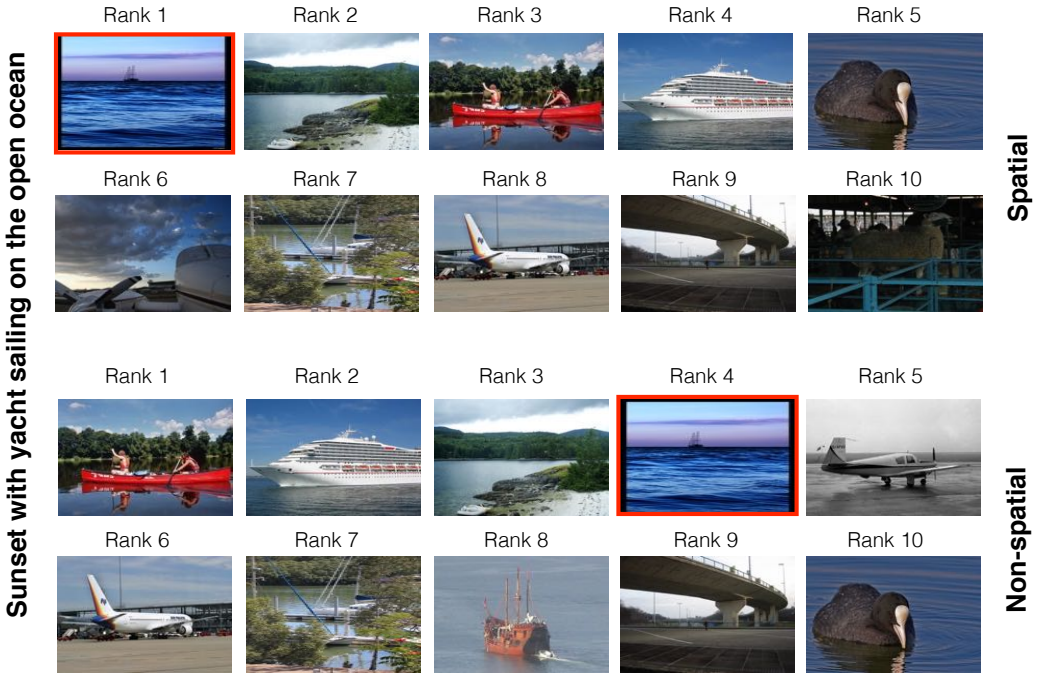


Figure 7.8: Image retrievals for a given query.

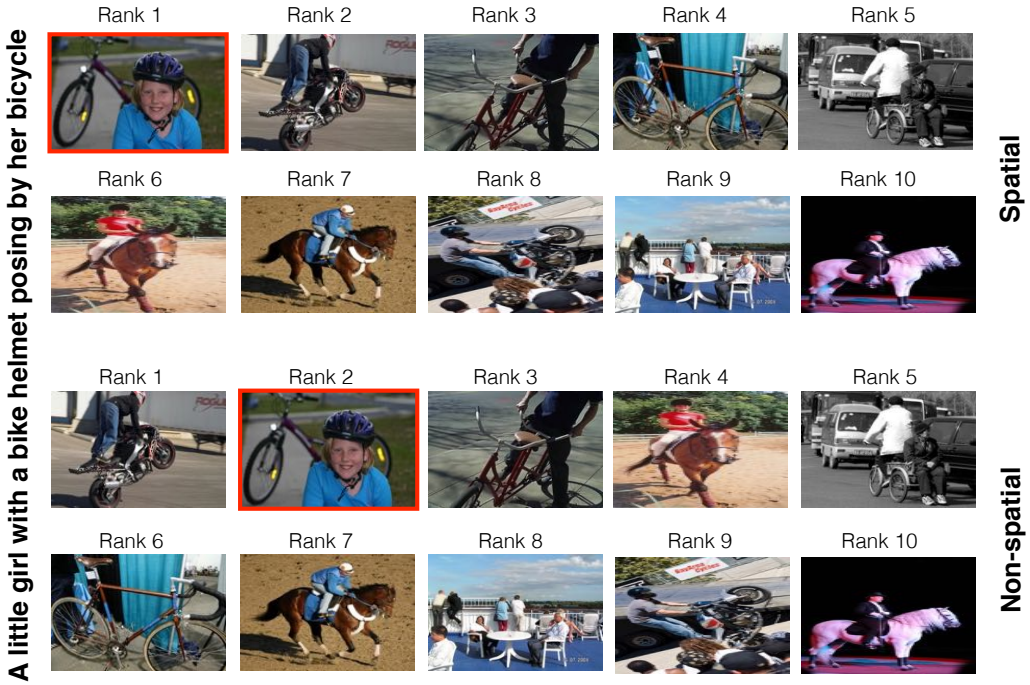


Figure 7.9: Image retrievals for a given query.

Figure 7.10: Image retrievals for a given query.

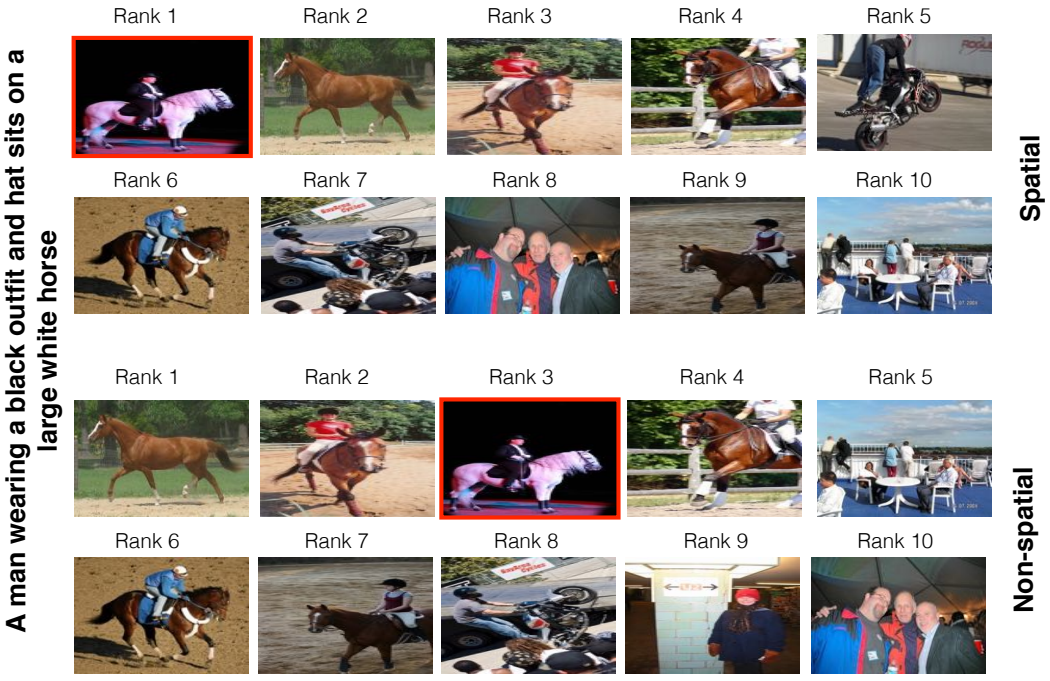


Figure 7.11: Image retrievals for a given query.

Chapter 8

Towards a Visual Turing Challenge

Contents

Progress in language and image understanding by machines – which we briefly covered in Chapter 1, Chapter 3, and Chapter 4 – has sparked the interest of the research community in more open-ended, holistic tasks such as the text-to-image retrieval task covered in Chapter 7. This progress has also refueled an old AI dream of building intelligent machines. We discuss a few prominent challenges that characterize such holistic tasks and argue for "question answering about images" as a particularly appealing instance of such a holistic task. In particular, we point out that it is a version of Turing Test that is likely to be more robust to over-interpretations and contrast it with tasks like grounding and generation of descriptions. Moreover, we discuss tools to measure progress in this field. A more concrete instantiation of the Visual Turing Test – along with a dataset, methods, and performance metrics – is later covered in Chapters 9, 10, and 11.

8.1 Introduction

I this section we argue for the task of answering to questions abour real-world images.

8.1.1 Towards a Visual Question Answering Task

Recently we witness a tremendous progress in the machine perception [Krizhevsky et al. 2012; Gupta et al. 2014; Girshick et al. 2014; Pishchulin et al. 2013; Tompson et al. 2014; He et al. 2014; Lee et al. 2014; Simonyan and Zisserman 2015] and in the language understanding [Blackburn and Bos 2005; Zettlemoyer and Collins 2007; Kwiatkowski et al. 2010; Mikolov et al. 2013; Cho et al. 2014] tasks. The progress in both fields has inspired researchers to build

holistic architectures for challenging grounding [Matuszek et al. 2012; Krishnamurthy and Kollar 2013], natural language generation from image/video [Farhadi et al. 2010; Kulkarni et al. 2011; Rohrbach et al. 2014], image-to-sentence alignment [Socher et al. 2014; Karpathy et al. 2014; Mao et al. 2014; Kong et al. 2014], and recently presented question-answering problems [Liang et al. 2013; Berant and Liang 2014; Iyyer et al. 2014; Fader et al. 2014; Malinowski and Fritz 2014a]. In this paper we argue for a Visual Turing Test - an open domain task of question-answering based on real-world images that resemblances the famous Turing Test [Turing 1950; LaCurts 2011] and deviates from other attempts [Shan et al. 2013; Lake et al. 2013; Battaglia et al. 2013] - and discuss challenges together with tools to benchmark different models on such task.

We typically measure the progress in the field by quantifying the performance of different methods against a carefully crafted set of benchmarks. Crowdsourcing in combination of machine learning approaches have served us well to generate curated datasets with a unique ground truth at scale [Welinder and Perona 2010; Welinder et al. 2010]. As the complexity and the openness of the task grows, the quest of crafting good benchmarks also becomes more difficult. First, interpreting and evaluating the answer of a system becomes increasingly difficult and ideally would rely on human judgement. Yet we want to have objective metrics that we can evaluate automatically at large scale. Second, establishing an evaluation methodology that assigns scores over a large output domain is challenging, as any system based on ontologies will have a limited coverage. Third, if our aim is to mimic human response, we have to deal with inherent ambiguities due to human judgement that stem from issues like binding, reference frames, social conventions. For instance, Malinowski and Fritz [2014a] report that for a question answering task on real-world images even human answers are inconsistent. Obviously this cannot be a problem of humans but rather argues for inherent ambiguities in the task.

Competing methods are validated against true annotations, but what is the 'truth" in a task where even human answers cannot completely agree with each other? Instead of seeking an unique, 'true" answer we suggest to look into 'social consensus' that takes multiple human answers as different interpretations of the question into account. This enables us to incorporate 'agreement' between the humans directly into the metric. Although the idea is not entirely new [Arbelaez et al. 2011; Hodosh et al. 2013; Farhadi et al. 2010], we believe it sits at the core of building more open and holistic challenges. The first implementations of the 'consensus' idea can be found in Malinowski et al. [2015], which is shown in Chapter 10. A similar idea is also used in the VQA challenge [Antol et al. 2015].

We exemplify some of our findings on the DAQUAR dataset [Malinowski and Fritz 2014a] with the aim of demonstrating different challenges that are present in the dataset. We hope that our exposition is helpful towards building a public Visual Turing Test and will generate a discussion for the agreeable evaluation procedure and designing systems that can address open domain tasks.

In this chapter, a holistic architecture (also a holistic learner) is a machine learning architecture designed to work on the task that fuses at least two modalities, e.g. language and vision. The external world is a part of a task accessible to the holistic learner only via sensors and it can be either a human world (the world that surrounds us), or a machine

world that models some aspects of the human world.

8.1.2 Why a Visual Turing Test?

Can machines answer questions about real-world images? Due to the increasing matureness of image and language understanding techniques, it seems a timely step to reach out for a challenging goal that combines the two and asks the question if systems that can answer questions on images is in reach. While this is an interesting scientific question and can advance image and language understanding, practical application for surveillance and assistance for the blind would directly follow.

Holistic task The task of answering questions about images implies a tight integration of two modalities - language and vision. The task demands a complete pipeline from interpretation of both modalities, finding a joint representation and inferring or deducing a coherent answer.

Focused task While there is a wide range of research tasks that aims at extracting semantic annotations from images, these always target certain aspects of the scene. By the introduction of a question about the scene, we basically parameterize the task by this input. Therefore, the question answering task can be seen as an open-ended task, which comes as an opportunity, but also as a challenge to define meaningful datasets. As we argue in Malinowski and Fritz [2015], as opposite to the famous Turing Test [Turing 1950] or an image captioning task [Vinyals et al. 2014], this proposed task should be less prone to over-interpretations by associating a meaning to machine answers or descriptions by a human interrogator.

Open-ended task The task that we are proposing does not explicitly constrain the space of possible questions, and hence presumably can be considered as an open-ended. At the same time, we can consider many variants of the answer space. On one hand, we can limit the answer space to K possible answers simplifying at the same time an automatic evaluation of the architectures on this task [Malinowski and Fritz 2014a]. On the other hand, we can consider truly open-ended answer space, where we expect from machines to produce answers reminding short, natural language, descriptions.

End-to-End task The skill of answering a question about an image, for sure requires some sort of scene and language understanding (Figure 8.1). But in contrast to traditional approaches that would build a system bottom up, the question answering allows methods to be to a certain extend agnostic to the internal representation, as no such intermediate steps are evaluated. By the merits of deep learning, such tasks can be trained end-to-end and internal representations can remain completely latent, yet their learning target implies the acquisition of certain competences in language and scene understanding. Methods that succeed in the task of answering questions about images have succeeded in scene and language understanding, but can go very different routes that need not to be compared.

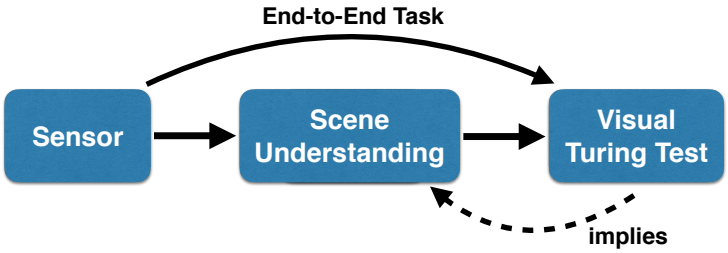


Figure 8.1: A good performance on a Visual Turing Test implies Scene Understanding. Yet, in contrast to many popular Image Understanding tasks, a Visual Turing Test is an end-to-end problem that doesn't evaluate how an image is represented.

Scalable annotation effort. As we strive to develop methods that understand visual scenes at increasing detail, the annotation effort becomes more and more laborious. In contrast, a question answering dataset can focus on particular aspects of a scene that go to a great level of detail, without the necessity to annotate 'everything".

Strategies for automatic evaluation While image captioning provides a great way to test scene comprehension and natural language generation, there is an issue of focus. Different human subjects may focus on different aspects of a scene and therefore multiple descriptions of a scene might be considered reasonable. This causes issues of evaluating such system. As the output is natural language, judging the quality of predictions require again a certain level of natural language understanding. Today's automatic evaluation metrics have their limitations. Therefore, while building DAQUAR [Malinowski and Fritz 2014a] we have deliberately decided against natural language answers in order to keep the evaluation tractable.

8.2 Challenges

As we strive for more holistic and open tasks such as grounding or question-answering based on images, we need to deal with a large gamut of challenges. In this section we have distilled and discuss some of the most prominent ones in order to guide the further discussion.

Vision and language *Scalability:* Perception and natural language understanding are crucial parts of holistic reasoning as they ground any representation in the external world and therefore serve as a common reference point for machines and humans. The human conceptualization divides these percepts into different instances, categories as well as spatio-temporal concepts. Architectures that aim at mimicking or reproducing this space of human concepts need to capture the same diversity and therefore scale up to thousands of concepts [Weston et al. 2011; Perronnin et al. 2012; Hoffman et al. 2014].
Concept ambiguity: As the number of categories grows, the semantic boundaries become more fuzzy, and hence ambiguities are inherently introduced [Lakoff 1990; Deng et al. 2010]. For instance, sometimes we may overlook the difference between 'night stand' and 'cabinet', or 'armchair' and 'sofa'. Therefore it is reasonable to expect from the holistic architectures

to create alternative hypotheses of the external world during inference. This also relates to the gradual category membership in human perception as portrayed in the prototype theory [Lakoff 1990; Rosch 1973].

Attributes: The human concepts are not limited to object categories, but also include attributes such as genders, colors, states (lights can be either on or off). Often these concepts cannot be learned on their own, but rather are contextualized by the associated noun. E.g. white in “white” elephant is surly different from “white” in white snow.

Ambiguity in reference resolution: Reliably answering on questions is challenging even for humans. The quality of an answer depends on how ambiguous and latent notions of reference frames and intentions are understood [Malinowski and Fritz 2014a; Golland et al. 2010]. Depending on the cultural bias and the context, we may use object-centric or observer-centric or even world-centric frames of reference [Levinson 2003]. Moreover, it is even unclear what 'with', 'beneath', 'over' mean. It seems at least difficult to symbolically define them in terms of predicates. While holistic learning and inference encompassing all the aforementioned aspects has yet to be shown, current research directions show promise [Beltagy et al. 2013; Rocktäschel et al. 2014; Lewis and Steedman 2014] by adapting the symbolic-based approaches [Zettlemoyer and Collins 2007; Kwiatkowski et al. 2010; Liang et al. 2013; Berant and Liang 2014] with vector-based approaches [Mikolov et al. 2013; Socher et al. 2014; Iyyer et al. 2014] to represent the meaning.

Common sense knowledge It turns out that some questions can solely be answered with the access to common sense knowledge with high reliability. For instance "Which object on the table is used for cutting?" already narrows the likely options significantly and the correct answer is probably “knife” or “scissors”. Other questions like "Which hand of the teacher is on her chin?" require the mixture of the vision and language. To understand the question, a holistic learner needs to first detect a person, figure out that the person may be a teacher, understand a gender of the person, detect her chin, understand 'left' and 'right' side, and finally relates 'her' with the 'teacher'.

However, different parts of the common sense knowledge can be used with different modality. An 'object for cutting' is not about seeing but about the affordance of the object and it cannot be learnt solely from the set of images. On the other hand things that often co-occur together may stand for the visual-based common sense knowledge. For instance we may expect to find a scissor or a pen inside a small plastic box, but never a wall or a window.

Common sense knowledge can help holistic machine learning architectures to either fulfill the task (question "Which object on the table is used for cutting?" can utilizes this type of knowledge), or limit the hypothesis space and hence to reduce the computational complexity of the search problem. For instance an architecture could be guided by its common sense knowledge to limit the space of possible locations of the 'scissors' and answer on "What is in front of scissors?" more effectively.

Defining a benchmark dataset and quantifying performance We argue that the question answering based on the visual input task significantly differ from the grounding

problem and has unique advantages towards defining a challenge dataset. Most prominently, the latter is about finding (either with a hand-crafted set of rules or learnt-based approaches) a mapping between the linguistic fragments and the physical world [Matuszek et al. 2012; Krishnamurthy and Kollar 2013; Harnad 1990], whereas the question answering task is about an end-to-end system where we do not necessarily want to enforce any constraints or penalty for the internal representation of the holistic learner. In this sense grounding is a latent sub-task that the holistic learner needs to solve, but will not be evaluated on. Finally, we argue that establishing benchmark dataset based on a question answering task similar to a turing test, is more tractable. Learning grounding asks for exhaustive symbolic-based annotations of the world, while question answering only needs textual annotations for the aspects that the question refers to.

8.3 DAQUAR: Building a Dataset for Visual Turing Challenge

DAQUAR [Malinowski and Fritz 2014a] is a challenging, large dataset for a question answering task based on real-world images. The images present real-world indoor scenes [Silberman et al. 2012], while the questions are unconstrained natural language sentences. DAQUAR's language scope is beyond the nouns or tuples that are typical to recognition datasets [Russakovsky et al. 2014; Rohrbach et al. 2011; Lan et al. 2012]. Other, linguistically rich datasets either do not tackle images at all [Zelle and Mooney 1996; Berant et al. 2013] or consider only few in very constrained domain [Krishnamurthy and Kollar 2013], or are more suitable for the learning an embedding/image-sentence retrieval or language generation [Kong et al. 2014; Rashtchian et al. 2010; Rohrbach et al. 2012; Gong et al. 2014]. In this section we discuss in isolation different challenges reflected in DAQUAR.

Vision and language The machine world in DAQUAR is represented as a set of images and questions about their content. DAQUAR contains 1088 different nouns in the question, 803 in the answers, and 1586 altogether (we use the Stanford POS Tagger [Toutanova et al. 2003] to extract the nouns from the questions). If we consider only nouns in singular form in the questions, we still have 573 categories. The current state-of-the-art semantic segmentation methods on the NYU-Depth V2 dataset [Silberman et al. 2012] can discriminate only between up to 37 object categories [Gupta et al. 2014; Lin et al. 2013; Gupta et al. 2013], much fewer to what is needed. DAQUAR also contains other parts of speech where only colors and spatial prepositions are grounded in Malinowski and Fritz [2014a].

Moreover, ambiguities naturally emerge due to fine grained categories that exist in DAQUAR. For instance 'night stand', 'stool' and 'cabinet' sometimes refer to the same thing. There is also a variation in the naming of colors among the annotations. Questions rely heavily on the spatial concepts with different frame of reference.

DAQUAR includes various challenges related to natural language understanding. Any semantic representation needs to work with the large number of predicates (reaching about 4 million to account different interpretations of the external world), with questions of

substantial length (10.5 words in average with variance 5.5; the longest question has 30 words), and possible language errors in the questions.

Common sense knowledge DAQUAR includes questions that can be reliably answered using common sense knowledge. For instance "Which object on the table is used for cutting?" already provides strong non-visual cues for the “cutting” object. Answers on other questions, such as "What is above the desk in front of scissors?", can be improved if the search space is reasonable restricted. Moreover, some annotators hypothesize missing parts of the object based on their common sense. To sum up, we believe that common sense knowledge is an interesting venue to explore with DAQUAR.

Question answering task The question answering task is also about understanding hidden intentions of the questioner with grounding as a sub-goal to solve. Some authors [Liang et al. 2013; Berant and Liang 2014; Malinowski and Fritz 2014a] treat the grounding (understood here as the logical representation of the meaning of the question) as a latent variable in the question answering task. Others [Golland et al. 2010] have modeled the pragmatic effects in the question answering task, but such approaches have never been shown to work in less constrained environments.

8.4 Quantifying the Performance of Holistic Architectures

Together with increasing complexity and openness of the task, quantifying performance of the holistic architectures becomes challenging due to several issues:
Automation: Evaluating answers on such complex tasks as answering on questions requires a quite deep understanding of natural language, involved concepts and hidden intentions of the questioner. The ideal but impractical metric would be to manually judge every single answer of every architecture individually. Since this is infeasible we are seeking an automatic approximation so that we can evaluate different holistic architectures at scale.
Ambiguity: The complex tasks that we are interested in are inherently ambiguous. The ambiguities stem from cultural bias, different frame of reference and fined grained categorization. This implies that multiple interpretations of a question are possible and hence many correct answers.
Coverage: Since there are multiple ways of expressing the same concept, the automatic performance metric should take the equivalence class among the answers into the consideration by assigning similar scores to all members of the same class. There are attempts to alleviate this issue via defining similarity scores [Wu and Palmer 1994] over the lexical databases [Miller 1995; Fellbaum 1999]. These approaches, however, lacks of coverage: we cannot assign a similarity between the terms that are not represented in the structure.

WUPS scores We exemplify the aforementioned requirements by illustrating the WUPS score - an automatic metric that quantifies performance of the holistic architectures proposed by Malinowski and Fritz [2014a]. This metric is motivated by the development of a ’soft’

generalization of accuracy that takes ambiguities of different concepts into account via the set membership measure μ:

$$\frac{1}{N}\sum_{i=1}^{N}\min\{\prod_{a\in A^i}\max_{t\in T^i}\mu(a,t),\ \prod_{t\in T^i}\max_{a\in A^i}\mu(a,t)\}\cdot 100 \tag{8.1}$$

where for each i-th question, A^i and T^i are the answers produced by the architecture and human respectively, and they are represented as bags of words. The authors of Malinowski and Fritz [2014a] have proposed using WUP similarity [Wu and Palmer 1994] as the membership measure μ in the WUPS score. Such choice of μ suffers from the aforementioned coverage problem and the whole metric takes only one human interpretation of the question into account.

Future directions for defining metrics Recent work provides several directions towards improving scores. To deal with ambiguities that stem from different readings of the same question we are collecting more human answers per question and we propose, based on that, two generalizations of WUPS score. The first, we call Interpretation Metric, runs Equation 8.1 over many human answers and takes the maximal score, so that the machine answer is high if it is similar to at least one human answer. However, with many human answers, we can also rank higher the machine answers that are 'socially agreeable' by measuring if they agree with most human answers. This can be done by averaging over multiple human answers. We call such second extension, Consensus Metric. The problem with coverage can be potentially alleviated with vector based representations [Mikolov et al. 2013] of the answers. Although in this case the coverage issues are less problematic, we understand the concerns that such score is dependent on the training data used to build such representation. On the other hand, due to abundance of textual data and recent improvements of vector based approaches [Mikolov et al. 2013; Pennington et al. 2014], we consider it as a valid alternative to similarities that are based on ontologies.

Experimental scenarios In many cases, success on challenging learning problems has been accelerated by use of external data in the training, e.g. in object detection [Girshick et al. 2014]. We believe that a Visual Turing challenge should consists of a sub-task with a prohibited use of auxiliary data to understand how the holistic learners generalize from limited and challenging data in a more established setup. On the other hand we should not limit ourselves to such artificial restrictions in building next generation of the holistic learners. Therefore open sub-tasks with a permissible use of another sources in the training have to be stated, including: additional vision and language resources, synthetic data and curated questions.

8.5 Summary

The goal of this contribution is to sparkle the discussions about benchmarking holistic architectures on complex and more open tasks. We identify particular challenges that holistic tasks should exhibit and exemplify how they are manifested in a recent question answering

challenge [Malinowski and Fritz 2014a]. To judge competing architectures and measure the progress on the task, we suggest several directions to further improve existing metrics, and discuss different experimental scenarios.

Chapter 9

A Multi-world Approach to Question Answering about Real-World Scenes based on Uncertain Input

Contents

Following the idea of creating a holistic task, described in Chapter 8, that does not evaluate on an internal representation of methods, allows for a scalable annotation effort, and feasible automatic performance metrics, this chapter introduces a concrete dataset, a method, and metrics for the "question answering about images" task. In this chapter, we introduce the first method for automatically answering questions about real-world indoor images by bringing together recent advances from natural language processing and computer vision. We combine discrete reasoning with uncertain predictions by a multi-world approach that represents uncertainty about the perceived world in a bayesian framework. Our approach to Visual Turing Test can handle human questions of high complexity about realistic scenes and replies with range of answer like counts, object classes, instances and lists of them. The system is directly trained from question-answer pairs. This chapter sets the basis that we later on, in Chapters 10 and 11, extend by introducing neural-based approaches to Visual Turing Test, along with more general performance metrics that account for uncertainties caused by various interpretations of a question and an image.

9.1 Introduction

As vision techniques like segmentation and object recognition begin to mature, there has been an increasing interest in broadening the scope of research to full scene understanding. But what is meant by 'understanding" of a scene and how do we measure the degree of 'understanding"? Most often 'understanding" refers to a correct labeling of pixels, regions or bounding boxes in terms of semantic annotations. All predictions made by such methods inevitably come with uncertainties attached due to limitations in features or data or even inherent ambiguity of the visual input.

Equally strong progress has been made on the language side, where methods have been proposed that can learn to answer questions solely from question-answer pairs [Liang et al. 2013]. These methods operate on a set of facts given to the system, which is refered to as a world. Based on that knowledge the answer is inferred by marginalizing over multiple interpretations of the question. However, the correctness of the facts is a core assumption.

We like to unite those two research directions by addressing a question answering task based on real-world images. To combine the probabilistic output of state-of-the-art scene segmentation algorithms, we propose a Bayesian formulation that marginalizes over multiple possible worlds that correspond to different interpretations of the scene.

To date, we are lacking a substantial dataset that serves as a benchmark for question answering on real-world images. Such a test has high demands on 'understanding" the visual input and tests a whole chain of perception, language understanding and deduction. This very much relates to the 'AI-dream" of building a turing test for vision. While we are still not ready to test our vision system on completely unconstrained settings that were envisioned in early days of AI, we argue that a question-answering task on complex indoor scenes is a timely step in this direction.

Contributions In this paper we combine automatic, semantic segmentations of real-world scenes with symbolic reasoning about questions in a Bayesian framework by proposing a multi-world approach for automatic question answering. We introduce a novel dataset of more than 12,000 question-answer pairs on RGBD images produced by humans, as a modern approach to a visual turing test. We benchmark our approach on this new challenge and show the advantages of our multi-world approach. Furthermore, we provide additional insights regarding the challenges that lie ahead of us by factoring out sources of error from different components.

9.2 Related work

Semantic parsers Our work is mainly inspired by Liang et al. [2013] that learns the semantic representation for the question answering task solely based on questions and answers in natural language. Although the architecture learns the mapping from weak supervision, it achieves comparable results to the semantic parsers that rely on manual annotations of logical forms [Kwiatkowski et al. 2010; Zettlemoyer and Collins 2007]. In contrast to our work, Liang et al. [2013] has never used the semantic parser to connect the

natural language to the perceived world.

Language and perception Previous work [Matuszek et al. 2012; Krishnamurthy and Kollar 2013] has proposed models for the language grounding problem with the goal of connecting the meaning of the natural language sentences to a perceived world. Both methods use images as the representation of the physical world, but concentrate rather on constrained domain with images consisting of very few objects. For instance Krishnamurthy and Kollar [2013] considers only two mugs, monitor and table in their dataset, whereas Matuszek et al. [2012] examines objects such as blocks, plastic food, and building bricks. In contrast, our work focuses on a diverse collection of real-world indoor RGBD images [Silberman et al. 2012] - with many more objects in the scene and more complex spatial relationship between them. Moreover, our paper considers complex questions - beyond the scope of Matuszek et al. [2012] and Krishnamurthy and Kollar [2013] - and reasoning across different images using only textual question-answer pairs for training. This imposes additional challenges for the question-answering engines such as scalability of the semantic parser, good scene representation, dealing with uncertainty in the language and perception, efficient inference and spatial reasoning. Although others [Kong et al. 2014; Karpathy et al. 2014] propose interesting alternatives for learning the language binding, it is unclear if such approaches can be used to provide answers on questions.

Integrated systems that execute commands Others [Matuszek et al. 2013; Levit and Roy 2007; Vogel and Jurafsky 2010; Tellex et al. 2011; Kruijff et al. 2007] focus on the task of learning the representation of natural language in the restricted setting of executing commands. In such scenario, the integrated systems execute commands given natural language input with the goal of using them in navigation. In our work, we aim for less restrictive scenario with the question-answering system in the mind. For instance, the user may ask our architecture about counting and colors ('How many green tables are in the image?'), negations ('Which images do not have tables?') and superlatives ('What is the largest object in the image?').

Probabilistic databases Similarly to Wick et al. [2010] that reduces Named Entity Recognition problem into the inference problem from probabilistic database, we sample multiple-worlds based on the uncertainty introduced by the semantic segmentation algorithm that we apply to the visual input.

9.3 Method

Our method answers on questions based on images by combining natural language input with output from visual scene analysis in a probabilistic framework as illustrated in Figure 9.1. In the single world approach, we generate a single perceived world $\mathcal{W}$ based on segmentations - a unique interpretation of a visual scene. In contrast, our multi-world approach integrates over many latent worlds $\mathcal{W}$, and hence taking different interpretations of the scene and question into account.

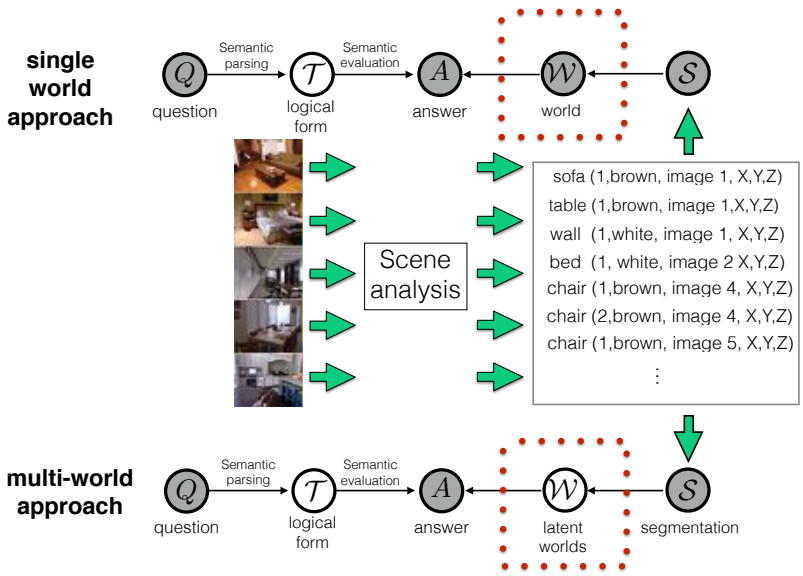


Figure 9.1: Overview of our approach to question answering with multiple latent worlds in contrast to single world approach.

Single-world approach for question answering problem We build on recent progress on end-to-end question answering systems that are solely trained on question-answer pairs (Q, A) [Liang et al. 2013]. Top part of Figure 9.1 outlines how we build on Liang et al. [2013] by modeling the logical forms associated with a question as latent variable $\mathcal{T}$ given a single world $\mathcal{W}$. More formally the task of predicting an answer $\mathcal{A}$ given a question $\mathcal{Q}$ and a world $\mathcal{W}$ is performed by computing the following posterior which marginalizes over the latent logical forms (semantic trees in Liang et al. [2013]) $\mathcal{T}$:

$$P(A|Q,\mathcal{W}) := \sum_{\mathcal{T}} P(A|\mathcal{T},\mathcal{W})P(\mathcal{T}|Q). \tag{9.1}$$

$P(A|\mathcal{T},\mathcal{W})$ corresponds to denotation of a logical form $\mathcal{T}$ on the world $\mathcal{W}$. In this setting, the answer is unique given the logical form and the world: $P(A|\mathcal{T},\mathcal{W}) = \mathbf{1}[A \in \sigma_{\mathcal{W}}(\mathcal{T})]$ with the evaluation function $\sigma_{\mathcal{W}}$, which evaluates a logical form on the world $\mathcal{W}$. Following Liang et al. [2013] we use DCS Trees that yield the following recursive evaluation function $\sigma_{\mathcal{W}}$: $\sigma_{\mathcal{W}}(\mathcal{T}) := \bigcap_j^d \{v \ : \ v \in \sigma_{\mathcal{W}}(p),\ t \in \sigma_{\mathcal{W}}(\mathcal{T}_j),\ \mathcal{R}_j(v,t)\}$ where $\mathcal{T} := \langle p, (\mathcal{T}_1,\mathcal{R}_1), (\mathcal{T}_2,\mathcal{R}_2), ..., (\mathcal{T}_d,\mathcal{R}_d)\rangle$ is the semantic tree with a predicate p associated with the current node, its subtrees $\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_d$, and relations $\mathcal{R}_j$ that define the relationship between the current node and a subtree $\mathcal{T}_j$.

In the predictions, we use a log-linear distribution $P(\mathcal{T}|Q) \propto \exp(\theta^T\phi(Q,\mathcal{T}))$ over the logical forms with a feature vector ϕ measuring compatibility between Q and $\mathcal{T}$ and parameters θ learnt from training data. Every component ϕ_j is the number of times that a specific feature template occurs in $(Q,\mathcal{T})$. We use the same templates as Liang et al. [2013]: string triggers a predicate, string is under a relation, string is under a trace predicate, two predicates are linked via relation and a predicate has a child. The model learns by alternating between searching over a restricted space of valid trees and gradient descent updates of the model parameters θ. We use the Datalog inference engine to produce the answers from

	Predicate	Definition
auxiliary relations	$closeAbove(A, B)$	$above(A, B)$ *and* $(Y_{min}(B) < Y_{max}(A) + \varepsilon)$
	$closeLeftOf(A, B)$	$leftOf(A, B)$ *and* $(X_{min}(B) < X_{max}(A) + \varepsilon)$
	$closeInFrontOf(A, B)$	$inFrontOf(A, B)$ *and* $(Z_{min}(B) < Z_{max}(A) + \varepsilon)$
	$X_{aux}(A, B)$	$X_{mean}(A) < X_{max}(B)$ *and* $X_{min}(B) < X_{mean}(A)$
	$Z_{aux}(A, B)$	$Z_{mean}(A) < Z_{max}(B)$ *and* $Z_{min}(B) < Z_{mean}(A)$
	$h_{aux}(A, B)$	$closeAbove(A, B)$ *or* $closeBelow(A, B)$
	$v_{aux}(A, B)$	$closeLeftOf(A, B)$ *or* $closeRightOf(A, B)$
	$d_{aux}(A, B)$	$closeInFrontOf(A, B)$ *or* $closeBehind(A, B)$
spatial	$leftOf(A, B)$	$X_{mean}(A) < X_{mean}(B))$
	$above(A, B)$	$Y_{mean}(A) < Y_{mean}(B)$
	$inFrontOf(A, B)$	$Z_{mean}(A) < Z_{mean}(B))$
	$on(A, B)$	$closeAbove(A, B)$ *and* $Z_{aux}(A, B)$ *and* $X_{aux}(A, B)$
	$close(A, B)$	$h_{aux}(A, B)$ *or* $v_{aux}(A, B)$ *or* $d_{aux}(A, B)$

Table 9.1: Predicates defining spatial relations between A and B. Auxiliary relations define actual spatial relations. The Y axis points downwards, functions $X_{max}, X_{min}, ...$ take appropriate values from the tuple *predicate*, and ε is a 'small' amount. Symmetrical relations such as $rightOf$, $below$, $behind$, etc. can readily be defined in terms of other relations (i.e. $below(A, B) = above(B, A)$).

the latent logical forms. The linguistic phenomena such as superlatives and negations are handled by the logical forms and the inference engine. For a detailed exposition, we refer the reader to Liang et al. [2013].

Question answering on real-world images based on a perceived world Similar to Krishnamurthy and Kollar [2013], we extend the work of Liang et al. [2013] to operate now on what we call *perceived world* $\mathcal{W}$. This still corresponds to the single world approach in our overview Figure 9.1. However our world is now populated with "facts" derived from automatic, semantic image segmentations $\mathcal{S}$. For this purpose, we build the world by running a state-of-the-art semantic segmentation algorithm [Gupta et al. 2013] over the images and collect the recognized information about objects such as object class, 3D position, and color [Van De Weijer et al. 2007] (Figure 9.1 - middle part). Every object hypothesis is therefore represented as an n-tuple: $predicate(instance\_id, image\_id, color, spatial\_loc)$ where $predicate \in \{bag, bed, books, ...\}$, $instance\_id$ is the object's id, $image\_id$ is id of the image containing the object, $color$ is estimated color of the object [Van De Weijer et al. 2007], and $spatial\_loc$ is the object's position in the image. Latter is represented as $(X_{min}, X_{max}, X_{mean}, Y_{min}, Y_{max}, Y_{mean}, Z_{min}, Z_{max}, Z_{mean})$ and defines minimal, maximal, and mean location of the object along X, Y, Z axes. To obtain the coordinates we fit axis parallel cuboids to the cropped 3d objects based on the semantic segmentation. Note that the X, Y, Z coordinate system is aligned with direction of gravity [Gupta et al. 2013]. As shown in Figure 9.2b, this is a more meaningful representation of the object's coordinates over simple image coordinates. The complete schema will be documented together with the code release.

We realize that the skilled use of spatial relations is a complex task and grounding spatial relations is a research thread on its own (e.g. Regier and Carlson [2001], Lan et al. [2012] and Guadarrama et al. [2013b]). For our purposes, we focus on predefined relations shown in Table 9.1, while the association of them as well as the object classes are still dealt within the question answering architecture.

Multi-worlds approach for combining uncertain visual perception and symbolic reasoning Up to now we have considered the output of the semantic segmentation as "hard facts", and hence ignored uncertainty in the class labeling. Every such labeling of the segments corresponds to different interpretation of the scene - different perceived world. Drawing on ideas from probabilistic databases [Wick et al. 2010], we propose a multi-world approach (Figure 9.1 - lower part) that marginalizes over multiple possible worlds $\mathcal{W}$ - multiple interpretations of a visual scene - derived from the segmentation $\mathcal{S}$. Therefore the posterior over the answer A given question Q and semantic segmentation S of the image marginalizes over the latent worlds $\mathcal{W}$ and logical forms $\mathcal{T}$:

$$P(A \mid Q, S) = \sum_{\mathcal{W}} \sum_{\mathcal{T}} P(A \mid \mathcal{W}, \mathcal{T}) P(\mathcal{W} \mid S)\ P(\mathcal{T} \mid Q) \tag{9.2}$$

The semantic segmentation of the image is a set of segments s_i with the associated probabilities p_{ij} over the C object categories c_j. More precisely $S = \{(s_1, L_1), (s_2, L_2), ..., (s_k, L_k)\}$ where $L_i = \{(c_j, p_{ij})\}_{j=1}^{C}$, $P(s_i = c_j) = p_{ij}$, and k is the number of segments of given image. Let $\hat{S}_f = \left\{(s_1, c_{f(1)}), (s_2, c_{f(2)}), ..., (s_k, c_{f(k)}))\right\}$ be an assignment of the categories into segments of the image according to the binding function $f \in \mathcal{F} = \{1, ..., C\}^{\{1,...,k\}}$. With such notation, for a fixed binding function f, a world $\mathcal{W}$ is a set of tuples consistent with $\hat{S}_f$, and define $P(W|S) = \prod_i p_{(i,f(i))}$. Hence we have as many possible worlds as binding functions, that is C^k. Equation 9.2 becomes quickly intractable for k and C seen in practice, wherefore we use a sampling strategy that draws a finite sample $\vec{\mathcal{W}} = (\mathcal{W}_1, \mathcal{W}_2, ..., \mathcal{W}_N)$ from $P(\cdot|S)$ under an assumption that for each segment s_i every object's category c_j is drawn independently according to p_{ij}. A few sampled perceived worlds are shown in Figure 9.2a.

Regarding the computational efficiency, computing $\sum_{\mathcal{T}} P(A \mid \mathcal{W}_i, \mathcal{T}) P(\mathcal{T} \mid Q)$ can be done independently for every $\mathcal{W}_i$, and therefore in parallel without any need for synchronization. Since for small N the computational costs of summing up computed probabilities is marginal, the overall cost is about the same as single inference modulo parallelism. The presented multi-world approach to question answering on real-world scenes is still an end-to-end architecture that is trained solely on the question-answer pairs.

Implementation and scalability For worlds containing many facts and spatial relations the induction step becomes computationally demanding as it considers all pairs of the facts (we have about 4 million predicates in the worst case). Therefore we use a batch-based approximation in such situations. Every image induces a set of facts that we call a batch of facts. For every test image, we find k nearest neighbors in the space of training batches with a boolean variant of TF.IDF to measure similarity [Manning et al. 2008]. This is equivalent to building a training world from k images with most similar content to the perceived world

of the test image. We use $k = 3$ and 25 worlds in our experiments. Dataset and the source code can be found in our website [1].

9.4 Experiments

9.4.1 DAQUAR

Images and Semantic Segmentation Our new dataset for question answering is built on top of the NYU-Depth V2 dataset [Silberman et al. 2012]. NYU-Depth V2 contains 1449 RGBD images together with annotated semantic segmentations (Figure 9.3) where every pixel is labeled into some object class with a confidence score. Originally 894 classes are considered. According to Gupta et al. [2013], we preprocess the data to obtain canonical views of the scenes and use X, Y, Z coordinates from the depth sensor to define spatial placement of the objects in 3D. To investigate the impact of uncertainty in the visual analysis of the scenes, we also employ computer vision techniques for automatic semantic segmentation. We use a state-of-the-art scene analysis method [Gupta et al. 2013] which maps every pixel into 40 classes: 37 informative object classes as well as 'other structure', 'other furniture' and 'other prop'. We ignore the latter three. We use the same data split as Gupta et al. [2013]: 795 training and 654 test images. To use our spatial representation on the image content, we fit 3d cuboids to the segmentations.

DAQUAR - the first dataset of question-answer pairs about real-world images In the spirit of a visual turing test, we collect question answer pairs from human annotators for the NYU dataset, and call it DAQUAR (DAtaset for Question Answering about Real-world images). In our work, we consider two types of the annotations: synthetic and human. The *synthetic question-answer pairs* are automatically generated question-answer pairs, which are based on the templates shown in Table 9.2. These templates are then instantiated with facts from the database. To collect 12468 *human question-answer pairs* we ask 5 in-house participants to provide questions and answers. They were instructed to give valid answers that are either basic colors [Van De Weijer et al. 2007], numbers or objects (894 categories) or sets of those. Besides the answers, we don't impose any constraints on the questions. We also

[1] https://www.d2.mpi-inf.mpg.de/visual-turing-challenge

	Description	Template	Example
Individual	counting	How many {object} are in {image_id}?	How many cabinets are in image1?
	counting and colors	How many {color} {object} are in {image_id}?	How many gray cabinets are in image1?
	room type	Which type of the room is depicted in {image_id}?	Which type of the room is depicted in image1?
	superlatives	What is the largest {object} in {image_id}?	What is the largest object in image1?
set	counting and colors	How many {color} {object}?	How many black bags?
	negations type 1	Which images do not have {object}?	Which images do not have sofa?
	negations type 2	Which images are not {room_type}?	Which images are not bedroom?
	negations type 3	Which images have {object} but do not have a {object}?	Which images have desk but do not have a lamp?

Table 9.2: Synthetic question-answer pairs. The questions can be about individual images or the sets of images.

don't correct the questions as we believe that the semantic parsers should be robust under the human errors. Finally, we use 6794 training and 5674 test question-answer pairs – about 9 pairs per image on average $(8.63, 8.75)$[2]. The database exhibit some biases showing humans tend to focus on a few prominent objects. For instance we have more than 400 occurrences of table and chair in the answers. In average the object's category occurs $(14.25, 4)$ times in training set and $(22.48, 5.75)$ times in total. Figure 9.4 shows example question-answer pairs together with the corresponding image that illustrate some of the challenges captured in this dataset.

Performance Measure While the quality of an answer that the system produces can be measured in terms of accuracy w.r.t. the ground truth (correct/wrong), we propose, inspired from the work on Fuzzy Sets [Zadeh 1965], a soft measure based on the WUP score [Wu and Palmer 1994], which we call WUPS (WUP Set) score. As the number of classes grows, the semantic boundaries between them are becoming more fuzzy. For example, both concepts 'carton' and 'box' have similar meaning, or 'cup' and 'cup of coffee' are almost indifferent. Therefore we seek a metric that measures the quality of an answer and penalizes naive solutions where the architecture outputs too many or too few answers. Standard Accuracy is defined as: $\frac{1}{N}\sum_{i=1}^{N}\mathbf{1}\{A^i = T^i\}\cdot 100$ where A^i, T^i are i-th answer and ground-truth respectively. Since both the answers may include more than one object, it is beneficial to represent them as sets of the objects $T = \{t_1, t_2, ...\}$. From this point of view we have for every $i \in \{1, 2, ..., N\}$:

$$\mathbf{1}\{A^i = T^i\} = \mathbf{1}\{A^i \subseteq T^i \cap T^i \subseteq A^i\} = \min\{\mathbf{1}\{A^i \subseteq T^i\},\ \mathbf{1}\{T^i \subseteq A^i\}\} \tag{9.3}$$

$$= \min\{\prod_{a\in A^i}\mathbf{1}\{a \in T^i\},\ \prod_{t\in T^i}\mathbf{1}\{t \in A^i\}\} \approx \min\{\prod_{a\in A^i}\mu(a \in T^i),\ \prod_{t\in T^i}\mu(t \in A^i)\} \tag{9.4}$$

We use a soft equivalent of the intersection operator in Equation 9.3, and a set membership measure μ, with properties $\mu(x \in X) = 1$ if $x \in X$, $\mu(x \in X) = \max_{y\in X}\mu(x = y)$ and $\mu(x = y) \in [0, 1]$, in Equation 9.4 with equality whenever $\mu = \mathbf{1}$. For μ we use a variant of Wu-Palmer similarity [Wu and Palmer 1994; Guadarrama et al. 2013a]. $\mathrm{WUP}(a, b)$ calculates similarity based on the depth of two words a and b in the taxonomy [Miller 1995; Fellbaum 1999], and define the WUPS score:

$$\mathrm{WUPS}(A, T) = \frac{1}{N}\sum_{i=1}^{N}\min\{\prod_{a\in A^i}\max_{t\in T^i}\mathrm{WUP}(a, t),\ \prod_{t\in T^i}\max_{a\in A^i}\mathrm{WUP}(a, t)\}\cdot 100 \tag{9.5}$$

Empirically, we have found that in our task a WUP score of around 0.9 is required for precise answers. Therefore we have implemented down-weighting $\mathrm{WUP}(a, b)$ by one order of magnitude $(0.1 \cdot \mathrm{WUP})$ whenever $\mathrm{WUP}(a, b) < t$ for a threshold t. We plot a curve over thresholds t ranging from 0 to 1 (Figure 9.5). Since 'WUPS at 0' refers to the most 'forgivable' measure without any down-weighting and 'WUPS at 1.0' corresponds to plain accuracy. Figure 9.5 benchmarks architectures by requiring answers with precision ranging from low to

[2] Our notation (x, y) denotes mean x and trimean y. We use Tukey's trimean $\frac{1}{4}(Q_1 + 2Q_2 + Q_3)$, where Q_j denotes the j-th quartile [Tukey 1977]. This measure combines the benefits of both median (robustness to the extremes) and empirical mean (attention to the hinge values).

high. Here we show some examples of the pure WUP score to give intuitions about the range: WUP(curtain, blinds) = 0.94, WUP(carton, box) = 0.94, WUP(stove, fire extinguisher) = 0.82.

9.4.2 Quantitative results

We perform a series of experiments to highlight particular challenges like uncertain segmentations, unknown true logical forms, some linguistic phenomena as well as show the advantages of our proposed multi-world approach. In particular, we distinguish between experiments on synthetic question-answer pairs (**SynthQA**) based on templates and those collected by annotators (**HumanQA**), automatic scene segmentation (**AutoSeg**) with a computer vision algorithm [Gupta et al. 2013] and human segmentations (**HumanSeg**) based on the ground-truth annotations in the NYU dataset as well as single world (**single**) and multi-world (**multi**) approaches.

9.4.2.1 Synthetic question-answer pairs (SynthQA)

Based on human segmentations (HumanSeg, 37 classes) (1st and 2nd rows in Table 9.3) uses automatically generated questions (we use templates shown in Table 9.2) and human segmentations. We have generated 20 training and 40 test question-answer pairs per template category, in total 140 training and 280 test pairs (as an exception negations type 1 and 2 have 10 training and 20 test examples each). This experiment shows how the architecture generalizes across similar type of questions provided that we have human annotation of the image segments. We have further removed negations of type 3 in the experiments as they have turned out to be particularly computationally demanding. Performance increases hereby from 56% to 59.9% with about 80% training Accuracy. Since some incorrect derivations give correct answers, the semantic parser learns wrong associations. Other difficulties stem from the limited training data and unseen object categories during training.

Based on automatic segmentations (AutoSeg, 37 classes, single) (3rd row in Table 9.3) tests the architecture based on uncertain facts obtained from automatic semantic segmentation [Gupta et al. 2013] where the most likely object labels are used to create a single world. Here, we are experiencing a severe drop in performance from 59.9% to 11.25% by switching from human to automatic segmentation. Note that there are only 37 classes available to us. This result suggests that the vision part is a serious bottleneck of the whole architecture.

Based on automatic segmentations using multi-world approach (AutoSeg, 37 classes, multi) (4th row in Table 9.3) shows the benefits of using our multiple worlds approach to predict the answer. Here we recover part of the lost performance by an explicit treatment of the uncertainty in the segmentations. Performance increases from 11.25% to 13.75%.

9.4.3 Human question-answer pairs (HumanQA)

Based on human segmentations 894 classes (HumanSeg, 894 classes) (1st row in Table 9.4) switching to human generated question-answer pairs. The increase in complexity is twofold. First, the human annotations exhibit more variations than the synthetic approach based on templates. Second, the questions are typically longer and include more spatially related objects. Figure 9.4 shows a few samples from our dataset that highlights challenges including complex and nested spatial reference and use of reference frames. We yield an accuracy of 7.86% in this scenario. As argued above, we also evaluate the experiments on the human data under the softer WUPS scores given different thresholds (Table 9.4 and Figure 9.5). In order to put these numbers in perspective, we also show performance numbers for two simple methods: predicting the most popular answer yields 4.4% Accuracy, and our untrained architecture gives 0.18% and 1.3% Accuracy and WUPS (at 0.9).

Based on human segmentations 37 classes (HumanSeg, 37 classes) (2nd row in Table 9.4) uses human segmentation and question-answer pairs. Since only 37 classes are supported by our automatic segmentation algorithm, we run on a subset of the whole dataset. We choose the 25 test images yielding a total of 286 question answer pairs for the following experiments. This yields 12.47% and 15.89% Accuracy and WUPS at 0.9 respectively.

Based on automatic segmentations (AutoSeg, 37 classes) (3rd row in Table 9.4) Switching from the human segmentations to the automatic yields again a drop from 12.47% to 9.69% in Accuracy and we observe a similar trend for the whole spectrum of the WUPS scores.

Based on automatic segmentations using multi-world approach (AutoSeg, 37 classes, multi) (4th row in Table 9.4) Similar to the synthetic experiments our proposed multi-world approach yields an improvement across all the measure that we investigate.

Human baseline (5th and 6th rows in Table 9.4 for 894 and 37 classes) shows human predictions on our dataset. We ask independent annotators to provide answers on the questions we have collected. They are instructed to answer with a number, basic colors [Van De Weijer et al. 2007], or objects (from 37 or 894 categories) or set of those. This performance gives a practical upper bound for the question-answering algorithms with an accuracy of 60.27% for the 37 class case and 50.20% for the 894 class case. We also ask to compare the answers of the AutoSeg single world approach with HumanSeg single world and AutoSeg multi-worlds methods. We use a two-sided binomial test to check if difference in preferences is statistically significant. As a result AutoSeg single world is the least preferred method with the p-value below 0.01 in both cases. Hence the human preferences are aligned with our accuracy measures in Table 9.4.

9.4.4 Qualitative results

We choose examples in Figure 9.6 to illustrate different failure cases - including last example where all methods fail. Since our multi-world approach generates different sets of facts

Synthetic question-answer pairs (SynthQA)

Segmentation	World(s)	# classes	Accuracy
HumanSeg	Single with Neg. 3	37	56.0%
HumanSeg	Single	37	59.5%
AutoSeg	Single	37	11.25%
AutoSeg	Multi	37	13.75%

Table 9.3: Accuracy results for the experiments with synthetic question-answer pairs.

about the perceived worlds, we observe a trend towards a better representation of high level concepts like 'counting' (leftmost the figure) as well as language associations. A substantial part of incorrect answers is attributed to missing segments, e.g. no pillow detection in third example in Figure 9.6.

9.5 Summary

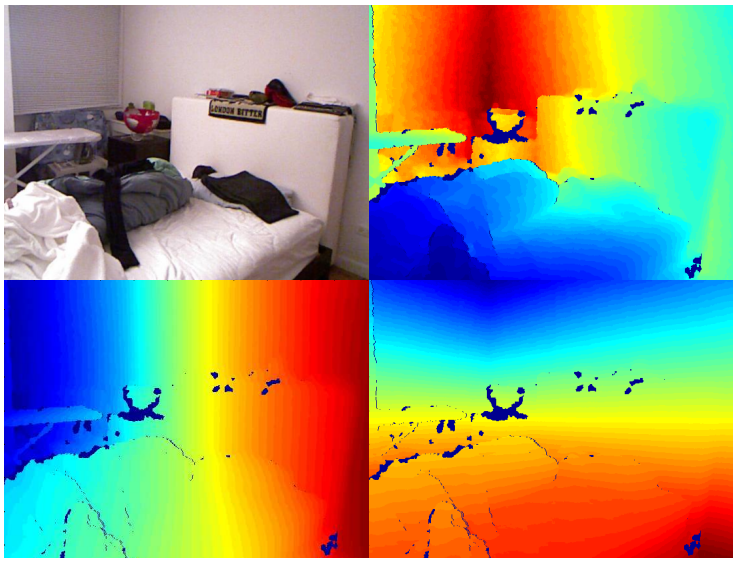
We propose a system and a dataset for question answering about real-world scenes that is reminiscent of a visual turing test. Despite the complexity in uncertain visual perception, language understanding and program induction, our results indicate promising progress in this direction. We bring ideas together from automatic scene analysis, semantic parsing with symbolic reasoning, and combine them under a multi-world approach. As we have mature techniques in machine learning, computer vision, natural language processing and deduction at our disposal, it seems timely to bring these disciplines together on this open challenge.

Human question-answer pairs (HumanQA)

Segmentation	World(s)	#classes	Accuracy	WUPS at 0.9	WUPS at 0
HumanSeg	Single	894	7.86%	11.86%	38.79%
HumanSeg	Single	37	12.47%	16.49%	50.28%
AutoSeg	Single	37	9.69%	14.73%	48.57%
AutoSeg	Multi	37	12.73%	18.10%	51.47%
Human Baseline		894	50.20%	50.82%	67.27%
Human Baseline		37	60.27%	61.04%	78.96%

Table 9.4: Accuracy and WUPS scores for the experiments with human question-answer pairs. We show WUPS scores at two opposite sides of the WUPS spectrum.

(a) Sampled worlds.



(b) Object's coordinates.

Figure 9.2: Figure 9.2a shows a few sampled worlds where only segments of the class 'person' are shown. In the clock-wise order: original picture, most confident world, and three possible worlds (gray-scale values denote the class confidence). Although, at first glance the most confident world seems to be a reasonable approach, our experiments show opposite - we can benefit from imperfect but multiple worlds. Figure 9.2b shows object's coordinates (original and Z, Y, X images in the clock-wise order), which better represent the spatial location of the objects than the image coordinates.

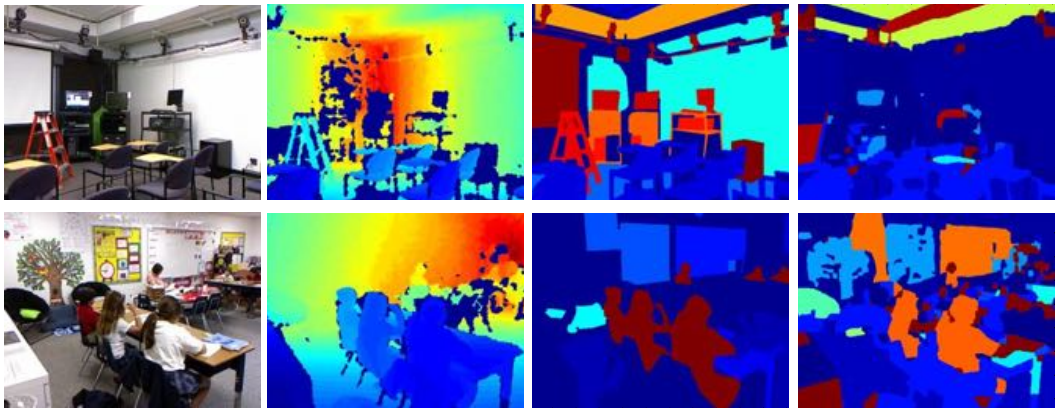


Figure 9.3: NYU-Depth V2 dataset: image, Z axis, ground truth and predicted semantic segmentations.

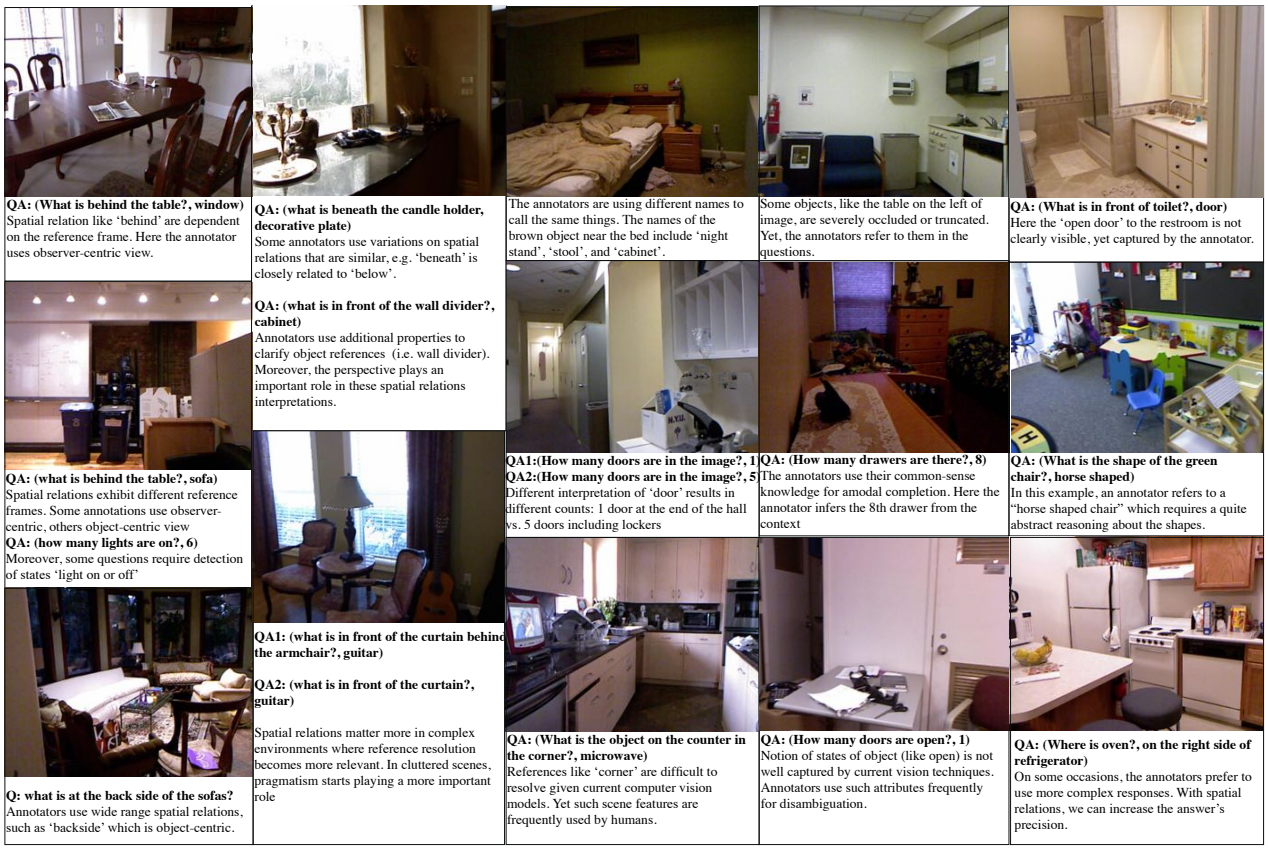


Figure 9.4: Examples of human generated question-answer pairs illustrating the associated challenges. In the descriptions we use following notation: 'A' - answer, 'Q' - question, 'QA' - question-answer pair. Last two examples (bottom-right column) are from the extended dataset not used in our experiments.

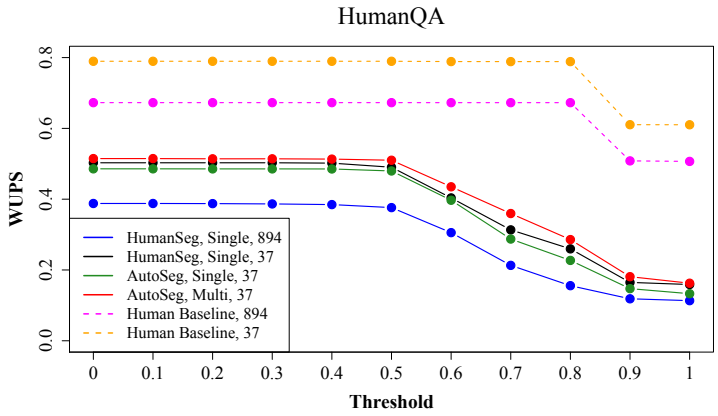


Figure 9.5: WUPS scores for different thresholds.

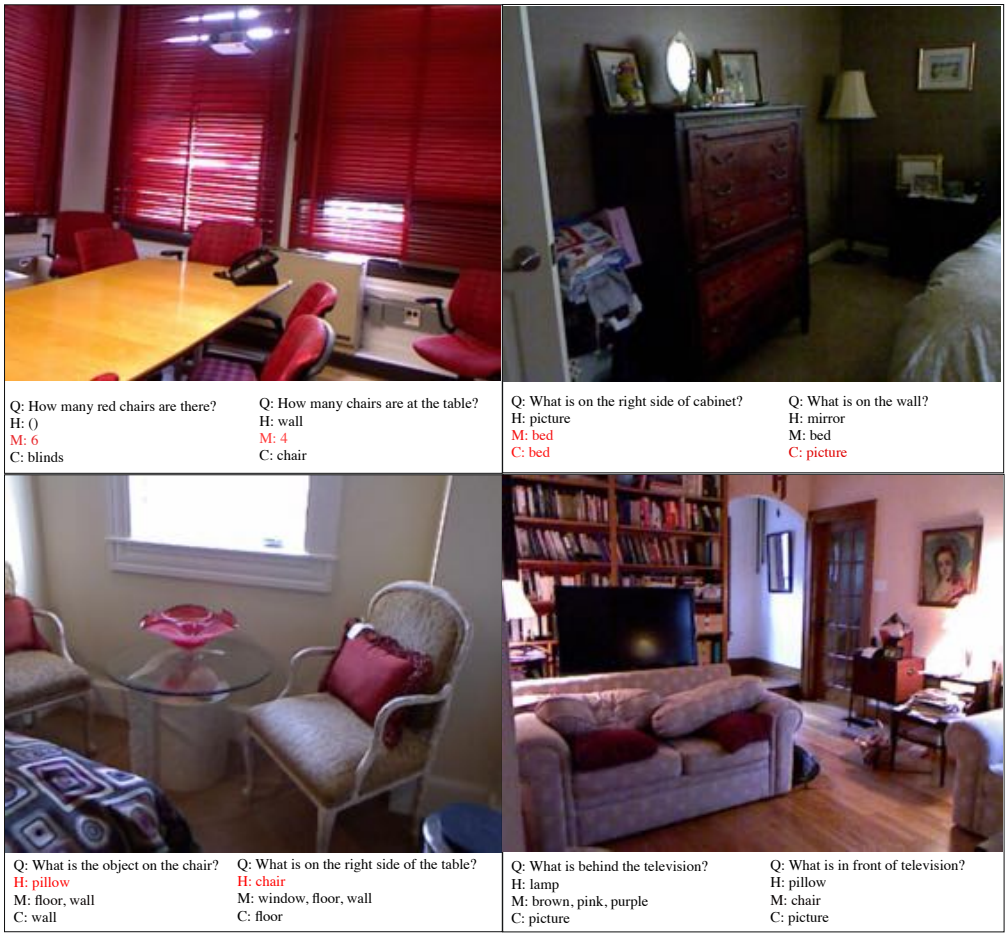


Figure 9.6: Questions and predicted answers. Notation: 'Q' - question, 'H' - architecture based on human segmentation, 'M' - architecture with multiple worlds, 'C' - most confident architecture, '()' - no answer. Red color denotes correct answer.

Chapter 10

Ask Your Neurons: A Neural-based Approach to Answering Questions about Images

Contents

In this chapter, we follow our main line of research on Visual Turing Test, which we started in Chapters 8, and 9. By combining latest advances in image representation and natural language processing, we propose Ask Your Neurons, an end-to-end formulation of this problem for which all parts are trained jointly. In contrast to previous efforts, we are facing a multi-modal problem where the language output (answer) is conditioned on visual and natural language input (question). Our result doubles the performance of our previous approach to Visual Turing Test (Chapter 9). Moreover, we also provide additional insights into the problem by analyzing how much information is contained only in the language part for which we provide a new human baseline. Further annotations were collected to study human consensus, which is related to the ambiguities inherent in this challenging task.

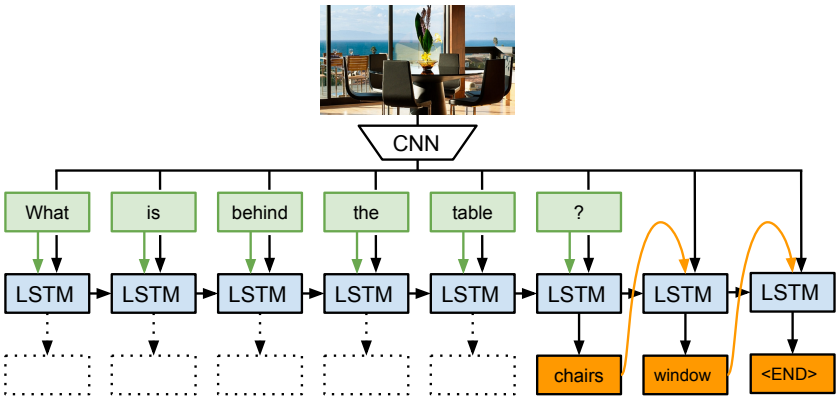


Figure 10.1: Our approach, Ask Your Neurons, to question answering with a Recurrent Neural Network using Long Short Term Memory (LSTM). To answer a question about an image, we feed in both, the image (CNN features) and the question (green boxes) into the LSTM. After the (variable length) question is encoded, we generate the answers (multiple words, orange boxes). During the answer generation phase the previously predicted answers are fed into the LSTM until the ⟨END⟩ symbol is predicted.

10.1 Introduction

With the advances of natural language processing and image understanding, more complex and demanding tasks have become within reach. Our aim is to take advantage of the most recent developments to push the state-of-the-art for answering natural language questions on real-world images. This task unites inference of question intends and visual scene understanding with a word sequence prediction task.

Most recently, architectures based on the idea of layered, end-to-end trainable artificial neural networks have improved the state of the art across a wide range of diverse tasks. Most prominently Convolutional Neural Networks have raised the bar on image classification tasks [Krizhevsky et al. 2012] and Long Short Term Memory Networks are dominating performance on a range of sequence prediction tasks such as machine translation [Sutskever et al. 2014].

Very recently these two trends of employing neural architectures have been combined fruitfully with methods that can generate image [Karpathy and Fei-Fei 2015] and video descriptions [Venugopalan et al. 2015a]. Both are conditioning on the visual features that stem from deep learning architectures and employ recurrent neural network approaches to produce descriptions.

To further push the boundaries and explore the limits of deep learning architectures, we propose an architecture for answering questions about images. In contrast to prior work, this task needs conditioning on language as well visual input. Both modalities have to be interpreted and jointly represented as an answer depends on inferred meaning of the question and image content.

While there is a rich body of work on natural language understanding that has addressed textual question answering tasks based on semantic parsing, symbolic representation and

deduction systems, which also has seen applications to question answering on images [Malinowski and Fritz 2014a], there is initial evidence that deep architectures can indeed achieve a similar goal [Weston et al. 2014]. This motivates our work to seek end-to-end architectures that learn to answer questions in a single holistic and monolithic model.

We propose Ask Your Neurons, an approach to question answering with a recurrent neural network. An overview is given in Figure 10.1. The image is analyzed via a Convolutional Neural Network (CNN) and the question together with the visual representation is fed into a Long Short Term Memory (LSTM) network. The system is trained to produce the correct answer to the question on the image. CNN and LSTM are trained jointly and end-to-end starting from words and pixels.

Contributions: We proposes a novel approach based on recurrent neural networks for the challenging task of answering of questions about images. It combines a CNN with a LSTM into an end-to-end architecture that predict answers conditioning on a question and an image. Our approach significantly outperforms prior work on this task – doubling the performance. We collect additional data to study human consensus on this task, propose two new metrics sensitive to these effects, and provide a new baseline, by asking humans to answer the questions without observing the image. We demonstrate a variant of our system that also answers question without accessing any visual information, which beats the human baseline.

10.2 Related Work

As our method touches upon different areas in machine learning, computer vision and natural language processing, we have organized related work in the following way:

Convolutional Neural Networks for visual recognition. We are building on the recent success of Convolutional Neural Networks (CNN) for visual recognition [Krizhevsky et al. 2012; LeCun et al. 1998b; Russakovsky et al. 2014], that are directly learnt from the raw image data and pre-trained on large image corpora. Due to the rapid progress in this area within the last two years, a rich set of models [Simonyan and Zisserman 2015; Szegedy et al. 2015] is at our disposal.

Recurrent Neural Networks (RNN) for sequence modeling. Recurrent Neural Networks allow Neural Networks to handle sequences of flexible length. A particular variant called Long Short Term Memory (LSTM) [Hochreiter and Schmidhuber 1997] has shown recent success on natural language tasks such as machine translation [Cho et al. 2014; Sutskever et al. 2014].

Combining RNNs and CNNs for description of visual content. The task of describing visual content like still images as well as videos has been successfully addressed with a combination of the previous two ideas [Donahue et al. 2015; Karpathy and Fei-Fei 2015; Venugopalan et al. 2015b; Vinyals et al. 2014; Zitnick et al. 2013]. This is achieved by using the RNN-type model that first gets to observe the visual content and is trained to

afterwards predict a sequence of words that is a description of the visual content. Our work extends this idea to question answering, where we formulate a model trained to generate an answer based on visual as well as natural language input.

Grounding of natural language and visual concepts. Dealing with natural language input does involve the association of words with meaning. This is often referred to as grounding problem - in particular if the "meaning" is associated with a sensory input. While such problems have been historically addressed by symbolic semantic parsing techniques [Krishnamurthy and Kollar 2013; Matuszek et al. 2012], there is a recent trend of machine learning-based approaches [Karpathy and Fei-Fei 2015; Karpathy et al. 2014; Kong et al. 2014] to find the associations. Our approach follows the idea that we do not enforce or evaluate any particular representation of "meaning" on the language or image modality. We treat this as latent and leave this to the joint training approach to establish an appropriate internal representation for the question answering task.

Textual question answering. Answering on purely textual questions has been studied in the NLP community [Berant and Liang 2014; Liang et al. 2013] and state of the art techniques typically employ semantic parsing to arrive at a logical form capturing the intended meaning and infer relevant answers. Only very recently, the success of the previously mentioned neural sequence models as RNNs has carried over to this task [Iyyer et al. 2014; Weston et al. 2014]. More specifically Iyyer et al. [2014] uses dependency-tree Recursive NN instead of LSTM, and reduce the question-answering problem to a classification task. Moreover, according to Iyyer et al. [2014] their method cannot be easily applied to vision. Weston et al. [2014] propose different kind of network - memory networks - and it is unclear how to apply Weston et al. [2014] to take advantage of the visual content. However, neither Iyyer et al. [2014] nor Weston et al. [2014] show an end-to-end, monolithic approaches that produce multiple words answers for question on images.

Visual Turing Test. Most recently several approaches have been proposed to approach Visual Turing Test [Malinowski and Fritz 2014b], i.e. answering questions about visual content. For instance Geman et al. [2015] have proposed a binary (yes/no) version of Visual Turing Test on synthetic data. In Malinowski and Fritz [2014a], we present a question answering system based on a semantic parser on a more varied set of human question-answer pairs. In contrast, in this work, our method is based on a neural architecture, which is trained end-to-end and therefore liberates the approach from any ontological commitment that would otherwise be introduced by a semantic parser.

We like to note that shortly after this work, several neural-based models [Ren et al. 2015a; Ma et al. 2015; Gao et al. 2015] have also been suggested. Also several new datasets for Visual Turing Tests have just been proposed [Antol et al. 2015; Yu et al. 2015a] that are worth further investigations.

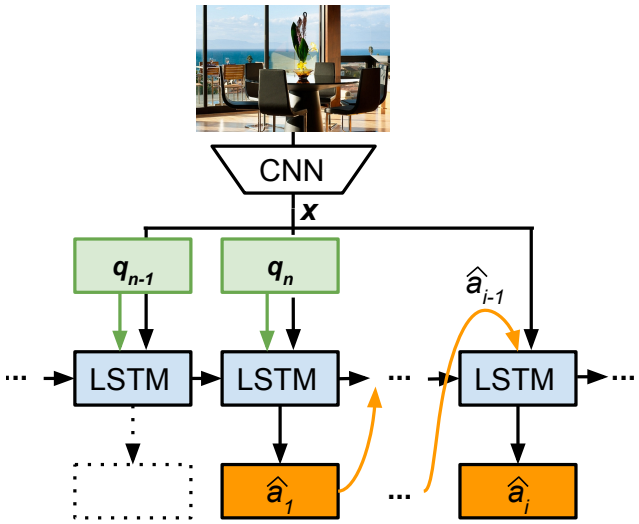


Figure 10.2: Our approach Ask Your Neurons, see Section 10.3 for details.

10.3 Approach

Answering questions on images is the problem of predicting an answer $\boldsymbol{a}$ given an image $\boldsymbol{x}$ and a question $\boldsymbol{q}$ according to a parametric probability measure:

$$\hat{\boldsymbol{a}} = \arg\max_{\boldsymbol{a} \in \mathcal{A}} p(\boldsymbol{a}|\boldsymbol{x}, \boldsymbol{q}; \boldsymbol{\theta}) \tag{10.1}$$

where $\boldsymbol{\theta}$ represent a vector of all parameters to learn and $\mathcal{A}$ is a set of all answers. Later we describe how we represent $\boldsymbol{x}$, $\boldsymbol{a}$, $\boldsymbol{q}$, and $p(\cdot|\boldsymbol{x}, \boldsymbol{q}; \boldsymbol{\theta})$ in more details.

In our scenario questions can have multiple word answers and we consequently decompose the problem to predicting a set of answer words $\boldsymbol{a}_{\boldsymbol{q},\boldsymbol{x}} = \{\boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_{\mathcal{N}(q,x)}\}$, where $\boldsymbol{a}_t$ are words from a finite vocabulary $\mathcal{V}'$, and $\mathcal{N}(q, x)$ is the number of answer words for the given question and image. In our approach, named Ask Your Neurons, we propose to tackle the problem as follows. To predict multiple words we formulate the problem as predicting a sequence of words from the vocabulary $\mathcal{V} := \mathcal{V}' \cup \{\$\}$ where the extra token \$ indicates the end of the answer sequence, and points out that the question has been fully answered. We thus formulate the prediction procedure recursively:

$$\hat{\boldsymbol{a}}_t = \arg\max_{\boldsymbol{a} \in \mathcal{V}} p(\boldsymbol{a}|\boldsymbol{x}, \boldsymbol{q}, \hat{A}_{t-1}; \boldsymbol{\theta}) \tag{10.2}$$

where $\hat{A}_{t-1} = \{\hat{\boldsymbol{a}}_1, \ldots, \hat{\boldsymbol{a}}_{\boldsymbol{t-1}}\}$ is the set of previous words, with $\hat{A}_0 = \{\}$ at the beginning, when our approach has not given any answer so far. The approach is terminated when $\hat{a}_t = \$$. We evaluate the method solely based on the predicted answer words ignoring the extra token \$. To ensure uniqueness of the predicted answer words, as we want to predict the set of answer words, the prediction procedure can be be trivially changed by maximizing over $\mathcal{V} \setminus \hat{A}_{t-1}$. However, in practice, our algorithm learns to not predict any previously predicted words.

As shown in Figure 10.1 and Figure 10.2, we feed Ask Your Neurons with a question as

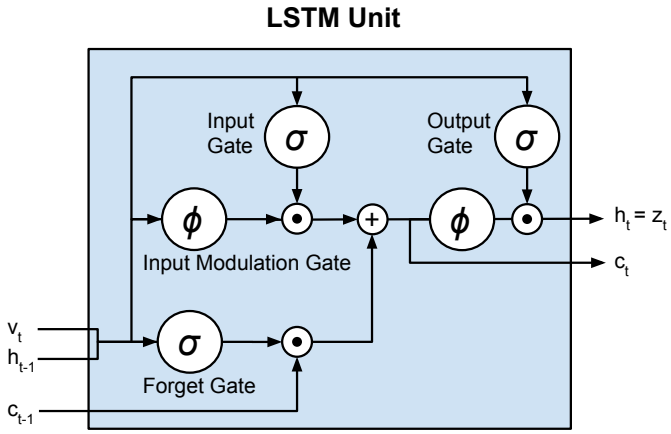


Figure 10.3: LSTM unit. See Section 10.3, Equations (11.1)-(11.6) for details.

a sequence of words, i.e. $\boldsymbol{q} = [\boldsymbol{q}_1, \ldots, \boldsymbol{q}_{n-1}, [\![?]\!]]$, where each $\boldsymbol{q}_t$ is the t-th word question and $[\![?]\!] := \boldsymbol{q}_n$ encodes the question mark - the end of the question. Since our problem is formulated as a variable-length input/output sequence, we model the parametric distribution $p(\cdot|\boldsymbol{x}, \boldsymbol{q}; \boldsymbol{\theta})$ of Ask Your Neurons with a recurrent neural network and a softmax prediction layer. More precisely, Ask Your Neurons is a deep network built of CNN [LeCun et al. 1998b] and Long-Short Term Memory (LSTM) [Hochreiter and Schmidhuber 1997]. LSTM has been recently shown to be effective in learning a variable-length sequence-to-sequence mapping [Donahue et al. 2015; Sutskever et al. 2014].

Both question and answer words are represented with one-hot vector encoding (a binary vector with exactly one non-zero entry at the position indicating the index of the word in the vocabulary) and embedded in a lower dimensional space, using a jointly learnt latent linear embedding. In the training phase, we augment the question words sequence $\boldsymbol{q}$ with the corresponding ground truth answer words sequence $\boldsymbol{a}$, i.e. $\hat{\boldsymbol{q}} := [\boldsymbol{q}, \boldsymbol{a}]$. During the test time, in the prediction phase, at time step t, we augment $\boldsymbol{q}$ with previously predicted answer words $\hat{\boldsymbol{a}}_{1..t} := [\hat{\boldsymbol{a}}_1, \ldots, \hat{\boldsymbol{a}}_{t-1}]$, i.e. $\hat{\boldsymbol{q}}_t := [\boldsymbol{q}, \hat{\boldsymbol{a}}_{1..t}]$. This means the question $\boldsymbol{q}$ and the previous answers are encoded implicitly in the hidden states of the LSTM, while the latent hidden representation is learnt. We encode the image $\boldsymbol{x}$ using a CNN and provide it at every time step as input to the LSTM. We set the input $\boldsymbol{v}_t$ as a concatenation of $[\boldsymbol{x}, \hat{\boldsymbol{q}}_t]$.

As visualized in detail in Figure 10.3, the LSTM unit takes an input vector $\boldsymbol{v}_t$ at each time step t and predicts an output word $\boldsymbol{z_t}$ which is equal to its latent hidden state $\boldsymbol{h}_t$. As discussed above $\boldsymbol{z_t}$ is a linear embedding of the corresponding answer word $\boldsymbol{a_t}$. In contrast to a simple RNN unit the LSTM unit additionally maintains a memory cell $\boldsymbol{c}$. This allows to learn long-term dynamics more easily and significantly reduces the vanishing and exploding gradients problem [Hochreiter and Schmidhuber 1997]. More precisely, we use the LSTM unit as described in Zaremba and Sutskever [2014] and the *Caffe* implementation from Donahue et al. [2015]. With the *sigmoid* nonlinearity $\sigma : \mathbb{R} \mapsto [0, 1]$, $\sigma(v) = (1 + e^{-v})^{-1}$ and the *hyperbolic tangent* nonlinearity $\phi : \mathbb{R} \mapsto [-1, 1]$, $\phi(v) = \frac{e^v - e^{-v}}{e^v + e^{-v}} = 2\sigma(2v) - 1$, the LSTM

updates for time step t given inputs $\boldsymbol{v}_t$, $\boldsymbol{h}_{t-1}$, and the memory cell $\boldsymbol{c}_{t-1}$ as follows:

$$\boldsymbol{i}_t = \sigma(W_{vi}\boldsymbol{v}_t + W_{hi}\boldsymbol{h}_{t-1} + \boldsymbol{b}_i) \tag{10.3}$$
$$\boldsymbol{f}_t = \sigma(W_{vf}\boldsymbol{v}_t + W_{hf}\boldsymbol{h}_{t-1} + \boldsymbol{b}_f) \tag{10.4}$$
$$\boldsymbol{o}_t = \sigma(W_{vo}\boldsymbol{v}_t + W_{ho}\boldsymbol{h}_{t-1} + \boldsymbol{b}_o) \tag{10.5}$$
$$\boldsymbol{g}_t = \phi(W_{vg}\boldsymbol{v}_t + W_{hg}\boldsymbol{h}_{t-1} + \boldsymbol{b}_g) \tag{10.6}$$
$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \boldsymbol{g}_t \tag{10.7}$$
$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \phi(\boldsymbol{c}_t) \tag{10.8}$$

where $\odot$ denotes element-wise multiplication. All the weights W and biases b of the network are learnt jointly with the cross-entropy loss. Conceptually, as shown in Figure 10.3, Equation 11.1 corresponds to the input gate, Equation 11.4 the input modulation gate, and Equation 11.2 the forget gate, which determines how much to keep from the previous memory c_{t-1} state. As Figures 10.1 and 10.2 suggest, all the output predictions that occur before the question mark are excluded from the loss computation, so that the model is penalized solely based on the predicted answer words.

Implementation We use default hyper-parameters of LSTM [Donahue et al. 2015] and CNN [Jia et al. 2014]. All CNN models are first pre-trained on the ImageNet dataset [Russakovsky et al. 2014], and next we randomly initialize and train the last layer together with the LSTM network on the task. We find this step crucial in obtaining good results. We have explored the use of a 2 layered LSTM model, but have consistently obtained worse performance. In a pilot study, we have found that *GoogleNet* architecture [Jia et al. 2014; Szegedy et al. 2015] consistently outperforms the *AlexNet* architecture [Jia et al. 2014; Krizhevsky et al. 2012] as a CNN model for our task and model.

10.4 Experiments

In this section we benchmark our method on a task of answering questions about images. We compare different variants of our proposed model to prior work in Section 10.4.1. In addition, in Section 10.4.2, we analyze how well questions can be answered without using the image in order to gain an understanding of biases in form of prior knowledge and common sense. We provide a new human baseline for this task. In Section 10.4.3 we discuss ambiguities in the question answering tasks and analyze them further by introducing metrics that are sensitive to these phenomena. In particular, the WUPS score [Malinowski and Fritz 2014a] is extended to a consensus metric that considers multiple human answers. Additional results are available in the supplementary material and on the project webpage [1].

Experimental protocol We evaluate our approach on the DAQUAR dataset [Malinowski and Fritz 2014a] which provides 12,468 human question answer pairs on images of indoor scenes [Silberman et al. 2012] and follow the same evaluation protocol by providing results on accuracy and the WUPS score at $\{0.9, 0.0\}$. We run experiments for the full dataset as

[1] https://www.d2.mpi-inf.mpg.de/visual-turing-challenge

	Accuracy	WUPS @0.9	WUPS @0.0
Malinowski et al. [Malinowski and Fritz 2014a]	7.86	11.86	38.79
Ask Your Neurons (ours)			
- multiple words	17.49	23.28	57.76
- single word	**19.43**	**25.28**	**62.00**
Human answers [Malinowski and Fritz 2014a]	50.20	50.82	67.27
Language only (ours)			
- multiple words	17.06	22.30	56.53
- single word	17.15	22.80	58.42
Human answers, no images	7.34	13.17	35.56

Table 10.1: Results on DAQUAR, all classes, single reference, in %.

well as their proposed reduced set that restricts the output space to only 37 object categories and uses 25 test images. In addition, we also evaluate the methods on different subsets of DAQUAR where only 1, 2, 3 or 4 word answers are present.

WUPS scores We base our experiments as well as the consensus metrics on WUPS scores [Malinowski and Fritz 2014a]. The metric is a generalization of the accuracy measure that accounts for word-level ambiguities in the answer words. For instance 'carton' and 'box' can be associated with a similar concept, and hence models should not be strongly penalized for this type of mistakes. Formally:

$$\text{WUPS}(A,T) = \frac{1}{N}\sum_{i=1}^{N} \min\{ \prod_{a\in A^i} \max_{t\in T^i} \mu(a,t), \prod_{t\in T^i} \max_{a\in A^i} \mu(a,t)\}$$

To embrace the aforementioned ambiguities, Malinowski and Fritz [2014a] suggest using a thresholded taxonomy-based Wu-Palmer similarity [Wu and Palmer 1994] for μ. The smaller the threshold the more forgiving metric. As in Malinowski and Fritz [2014a], we report WUPS at two extremes, 0.0 and 0.9.

10.4.1 Evaluation of Ask Your Neurons

We start with the evaluation of our Ask Your Neurons on the full DAQUAR dataset in order to study different variants and training conditions. Afterwards we evaluate on the reduced DAQUAR for additional points of comparison to prior work.

Results on full DAQUAR Table 10.1 shows the results of our Ask Your Neurons method on the full set ("multiple words") with 653 images and 5673 question-answer pairs available

	Accuracy	WUPS @0.9	WUPS @0.0
Ask Your Neurons (ours)	21.67	27.99	65.11
Language only (ours)	19.13	25.16	61.51

Table 10.2: Results of the single word model on the one-word answers subset of DAQUAR, all classes, single reference, in %.

at test time. In addition, we evaluate a variant that is trained to predict only a single word ("single word") as well as a variant that does not use visual features ("Language only"). In comparison to the prior work [Malinowski and Fritz 2014a] (shown in the first row in Table 10.1), we observe strong improvements of over 9% points in accuracy and over 11% in the WUPS scores [second row in Table 10.1 that corresponds to "multiple words"]. Note that, we achieve this improvement despite the fact that the only published number available for the comparison on the full set uses ground truth object annotations [Malinowski and Fritz 2014a] – which puts our method at a disadvantage. Further improvements are observed when we train only on a single word answer, which doubles the accuracy obtained in prior work. We attribute this to a joint training of the language and visual representations and the dataset bias, where about 90% of the answers contain only a single word.

We further analyze this effect in Figure 10.4, where we show performance of our approach ("multiple words") in dependence on the number of words in the answer (truncated at 4 words due to the diminishing performance). The performance of the "single word" variants on the one-word subset are shown as horizontal lines. Although accuracy drops rapidly for longer answers, our model is capable of producing a significant number of correct two words answers. The "single word" variants have an edge on the single answers and benefit from the dataset bias towards these type of answers. Quantitative results of the "single word" model on the one-word answers subset of DAQUAR are shown in Table 10.2. While we have made substantial progress compared to prior work, there is still a 30% points margin to human accuracy and 25 in WUPS score ["Human answers" in Table 10.1].

Results on reduced DAQUAR In order to provide performance numbers that are comparable to the proposed Multi-World approach in Malinowski and Fritz [2014a], we also run our method on the reduced set with 37 object classes and only 25 images with 297 question-answer pairs at test time.

Table 10.3 shows that Ask Your Neurons also improves on the reduced DAQUAR set, achieving 34.68% Accuracy and 40.76% WUPS at 0.9 substantially outperforming Malinowski and Fritz [2014a] by 21.95% Accuracy and 22.6 WUPS. Similarly to previous experiments, we achieve the best performance using the "single word" variant.

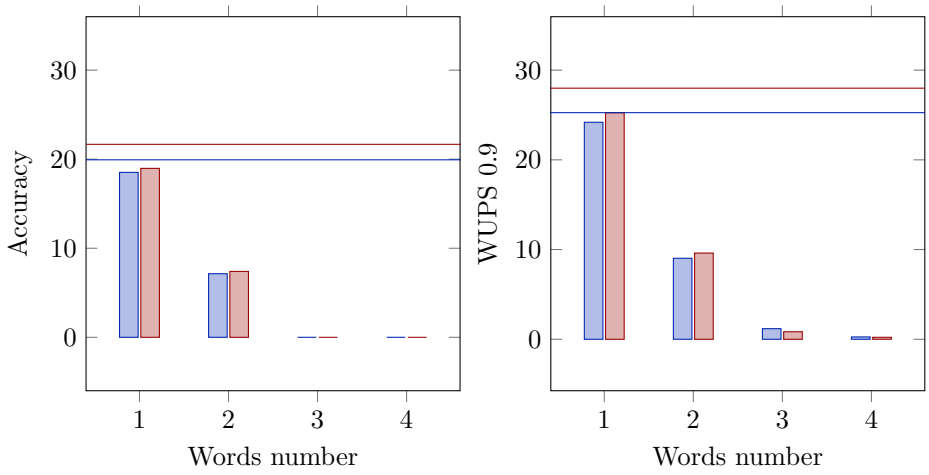


Figure 10.4: Language only (blue bar) and Ask Your Neurons (red bar) "multi word" models evaluated on different subsets of DAQUAR. We consider 1, 2, 3, 4 word subsets. The blue and red horizontal lines represent "single word" variants evaluated on the answers with exactly 1 word.

10.4.2 Answering questions without looking at images

In order to study how much information is already contained in questions, we train a version of our model that ignores the visual input. The results are shown in Table 10.1 and Table 10.3 under "Language only (ours)". The best "Language only" models with 17.15% and 32.32% compare very well in terms of accuracy to the best models that include vision. The latter achieve 19.43% and 34.68% on the full and reduced set respectively.

In order to further analyze this finding, we have collected a new human baseline "Human answer, no image", where we have asked participants to answer on the DAQUAR questions without looking at the images. It turns out that humans can guess the correct answer in 7.86% of the cases by exploiting prior knowledge and common sense. Interestingly, our best "language only" model outperforms the human baseline by over 9%. A substantial number of answers are plausible and resemble a form of common sense knowledge employed by humans to infer answers without having seen the image.

10.4.3 Human Consensus

We observe that in many cases there is an inter human agreement in the answers for a given image and question and this is also reflected by the human baseline performance on the question answering task of 50.20% ["Human answers" in Table 10.1]. We study and analyze this effect further by extending our dataset to multiple human reference answers in Section 10.4.3.1, and proposing a new measure – inspired by the work in psychology [Cohen et al. 1960; Fleiss and Cohen 1973; Nakashole et al. 2013] – that handles disagreement in Section 10.4.3.2, as well as conducting additional experiments in Section 10.4.3.3.

	Accu-racy	WUPS @0.9	WUPS @0.0
Malinowski et al. [Malinowski and Fritz 2014a]	12.73	18.10	51.47
Ask Your Neurons (ours)			
- multiple words	29.27	36.50	79.47
- single word	**34.68**	**40.76**	79.54
Language only (ours)			
- multiple words	32.32	38.39	80.05
- single word	31.65	38.35	**80.08**

Table 10.3: Results on reduced DAQUAR, single reference, with a reduced set of 37 object classes and 25 test images with 297 question-answer pairs, in %

10.4.3.1 DAQUAR-Consensus

In order to study the effects of consensus in the question answering task, we have asked multiple participants to answer the same question of the DAQUAR dataset given the respective image. We follow the same scheme as in the original data collection effort, where the answer is a set of words or numbers. We do not impose any further restrictions on the answers. This extends the original data [Malinowski and Fritz 2014a] to an average of 5 test answers per image and question. We refer to this dataset as DAQUAR-Consensus.

10.4.3.2 Consensus Measures

While we have to acknowledge inherent ambiguities in our task, we seek a metric that prefers an answer that is commonly seen as preferred. We make two proposals:

Average Consensus: We use our new annotation set that contains multiple answers per question in order to compute an expected score in the evaluation:

$$\frac{1}{NK}\sum_{i=1}^{N}\sum_{k=1}^{K}\min\{\prod_{a\in A^i}\max_{t\in T_k^i}\mu(a,t),\ \prod_{t\in T_k^i}\max_{a\in A^i}\mu(a,t)\} \tag{10.9}$$

where for the i-th question A^i is the answer generated by the architecture and T_k^i is the k-th possible human answer corresponding to the k-th interpretation of the question. Both answers A^i and T_k^i are sets of the words, and μ is a membership measure, for instance WUP [Wu and Palmer 1994]. We call this metric "Average Consensus Metric (ACM)" since, in the limits, as K approaches the total number of humans, we truly measure the inter human agreement of every question.

Min Consensus: The Average Consensus Metric puts more weights on more "mainstream" answers due to the summation over possible answers given by humans. In order to measure if the result was at least with one human in agreement, we propose a "Min Consensus Metric

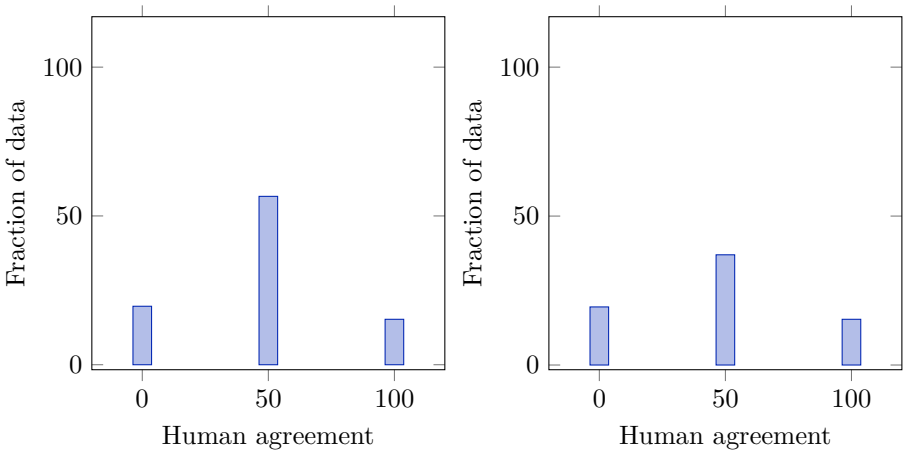


Figure 10.5: Study of inter human agreement. At x-axis: no consensus (0%), at least half consensus (50%), full consensus (100%). Results in %. Left: consensus on the whole data, right: consensus on the test data.

(MCM)" by replacing the averaging in Equation 10.9 with a max operator. We call such metric Min Consensus and suggest using both metrics in the benchmarks. We will make the implementation of both metrics publicly available.

$$\frac{1}{N}\sum_{i=1}^{N}\max_{k=1}^{K}\left(\min\{\prod_{a\in A^i}\max_{t\in T_k^i}\mu(a,t),\prod_{t\in T_k^i}\max_{a\in A^i}\mu(a,t)\}\right) \tag{10.10}$$

Intuitively, the max operator uses in evaluation a human answer that is the closest to the predicted one – which represents a minimal form of consensus.

10.4.3.3 Consensus results

Using the multiple reference answers in DAQUAR-Consensus we can show a more detailed analysis of inter human agreement. Figure 10.5 shows the fraction of the data where the answers agree between all available questions ("100"), at least 50% of the available questions and do not agree at all (no agreement - "0"). We observe that for the majority of the data, there is a partial agreement, but even full disagreement is possible. We split the dataset into three parts according to the above criteria "No agreement", "$\geq$ 50% agreement" and "Full agreement" and evaluate our models on these splits (Table 10.4 summarizes the results). On subsets with stronger agreement, we achieve substantial gains of up to 10% and 20% points in accuracy over the full set (Table 10.1) and the **Subset: No agreement** (Table 10.4), respectively. These splits can be seen as curated versions of DAQUAR, which allows studies with factored out ambiguities.

The aforementioned "Average Consensus Metric" generalizes the notion of the agreement, and encourages predictions of the most agreeable answers. On the other hand "Min Consensus Metric" has a desired effect of providing a more optimistic evaluation. Table 10.5 shows the application of both measures to our data and models.

	Accuracy	WUPS @0.9	WUPS @0.0
Subset: No agreement			
Language only (ours)			
- multiple words	8.86	12.46	38.89
- single word	8.50	12.05	40.94
Ask Your Neurons (ours)			
- multiple words	**10.31**	**13.39**	40.05
- single word	9.13	13.06	**43.48**
Subset: $\geq 50\%$ agreement			
Language only (ours)			
- multiple words	21.17	27.43	66.68
- single word	20.73	27.38	67.69
Ask Your Neurons (ours)			
- multiple words	20.45	27.71	67.30
- single word	**24.10**	**30.94**	**71.95**
Subset: Full Agreement			
Language only (ours)			
- multiple words	27.86	35.26	78.83
- single word	25.26	32.89	79.08
Ask Your Neurons (ours)			
- multiple words	22.85	33.29	78.56
- single word	**29.62**	**37.71**	**82.31**

Table 10.4: Results on DAQUAR, all classes, single reference in % (the subsets are chosen based on DAQUAR-Consensus).

	Accuracy	WUPS @0.9	WUPS @0.0
Average Consensus Metric			
Language only (ours)			
- multiple words	11.60	18.24	52.68
- single word	11.57	18.97	54.39
Ask Your Neurons (ours)			
- multiple words	11.31	18.62	53.21
- single word	**13.51**	**21.36**	**58.03**
Min Consensus Metric			
Language only (ours)			
- multiple words	22.14	29.43	66.88
- single word	22.56	30.93	69.82
Ask Your Neurons (ours)			
- multiple words	22.74	30.54	68.17
- single word	**26.53**	**34.87**	**74.51**

Table 10.5: Results on DAQUAR-Consensus, all classes, consensus in %.

Moreover, Table 10.6 shows that "MCM" applied to human answers at test time captures ambiguities in interpreting questions by improving the score of the human baseline from Malinowski and Fritz [2014a] (here, as opposed to Table 10.5, we exclude the original human answers from the measure). It also cooperates well with WUPS at 0.9, which takes word ambiguities into account, gaining an about 20% higher score.

10.4.4 Qualitative results

We show predicted answers of different variants of our architecture in Table 10.7, 10.8, and 10.9. We have chosen the examples to highlight differences between Ask Your Neurons and the "Language only". We use a "multiple words" approach only in Table 10.8, otherwise the "single word" model is shown. Despite some failure cases, "Language only" makes "reasonable guesses" like predicting that the largest object could be table or an object that could be found on the bed is either a pillow or doll.

10.4.5 Failure cases

While our method answers correctly on a large part of the challenge (e.g. ≈ 35 WUPS at 0.9 on "what color" and "how many" question subsets), spatial relations (≈ 21 WUPS at 0.9) which account for a substantial part of DAQUAR remain challenging. Other errors involve questions with small objects, negations, and shapes (below 12 WUPS at 0.9). Too few training data points for the aforementioned cases may contribute to these mistakes.

Table 10.9 shows examples of failure cases that include (in order) strong occlusion, a

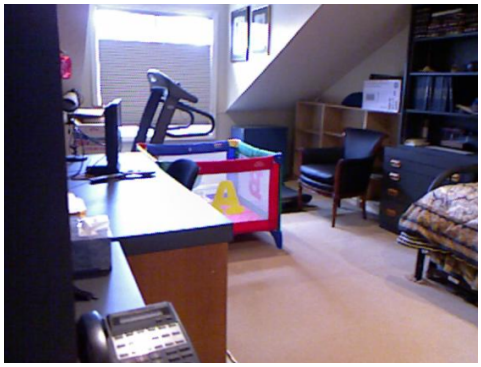
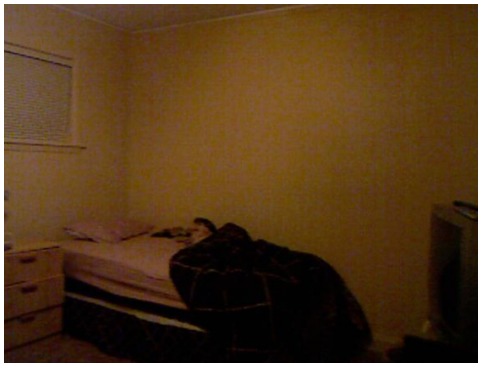
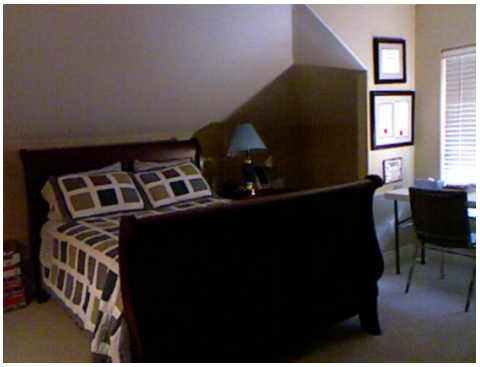
	Accuracy	WUPS @0.9	WUPS @0.0
WUPS [Malinowski and Fritz 2014a]	50.20	50.82	67.27
ACM (ours)	36.78	45.68	64.10
MCM (ours)	60.50	69.65	82.40

Table 10.6: Min and Average Consensus on human answers from DAQUAR, as reference sentence we use all answers in DAQUAR-Consensus which are not in DAQUAR, in %

possible answer not captured by our ground truth answers, and unusual instances (red toaster).

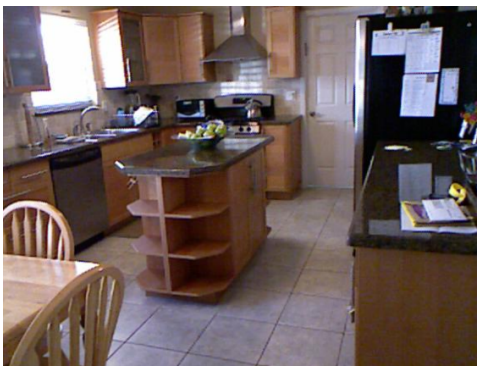
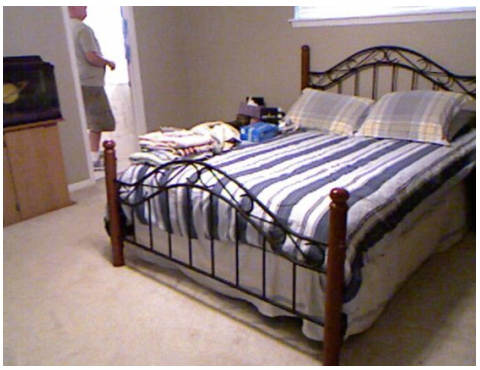
10.5 Conclusions

We have presented a neural architecture for answering natural language questions about images that contrasts with prior efforts based on semantic parsing and outperforms prior work by doubling performance on this challenging task. A variant of our model that does not use the image to answer the question performs only slightly worse and even outperforms a new human baseline that we have collected under the same condition. We conclude that our model has learnt biases and patterns that can be seen as forms of common sense and prior knowledge that humans use to accomplish this task. We observe that indoor scene statistics, spatial reasoning, and small objects are not well captured by the global CNN representation, but the true limitations of this representation can only be explored on larger datasets. We extended our existing DAQUAR dataset to DAQUAR-Consensus, which now provides multiple reference answers which allows to study inter-human agreement and consensus on the question answer task. We propose two new metrics: “Average Consensus”, which takes into account human disagreement, and “Min Consensus” that captures disagreement in human question answering.

	What is on the right side of the cabinet?	How many drawers are there?	What is the largest object?
Ask Your Neurons:	bed	3	bed
Language only:	bed	6	table

Table 10.7: Examples of questions and answers. Correct predictions are colored in green, incorrect in red.


	What is on the refrigerator?	What is the colour of the comforter?	What objects are found on the bed?
Ask Your Neurons:	magnet, paper	blue, white	bed sheets, pillow
Language only:	magnet, paper	blue, green, red, yellow	doll, pillow

Table 10.8: Examples of questions and answers with multiple words. Correct predictions are colored in green, incorrect in red.

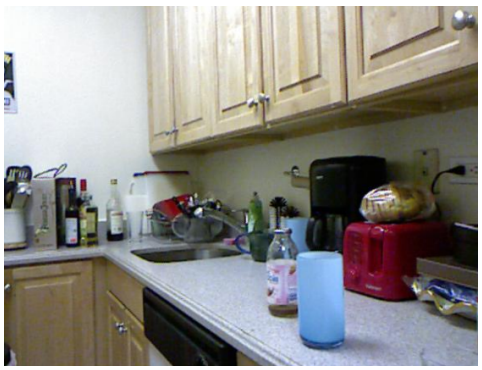
	 How many chairs are there?	 What is the object fixed on the window?	 Which item is red in colour?
Ask Your Neurons:	1	curtain	remote control
Language only:	4	curtain	clock
Ground truth answers:	2	handle	toaster

Table 10.9: Examples of questions and answers - failure cases.

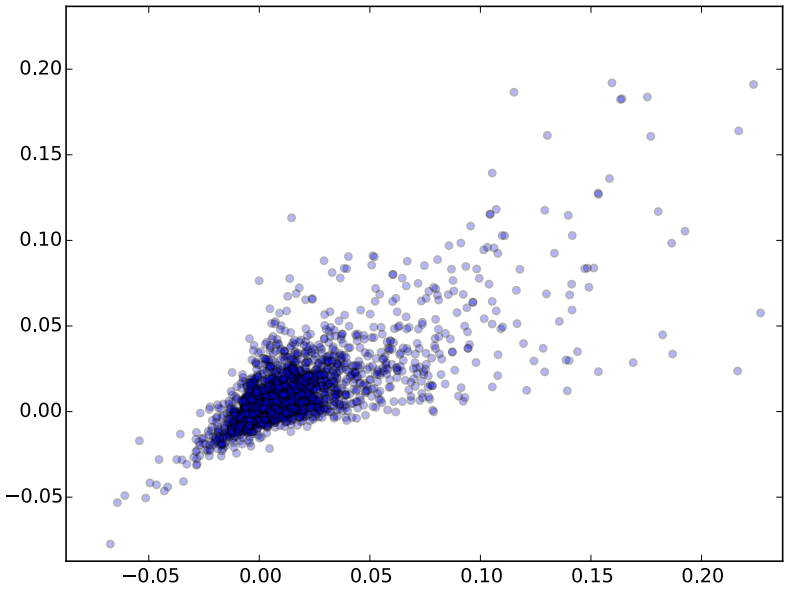


Figure 10.6: Figure showing correlation between question and answer words of the 'Language only' model (at x-axis), and a similar correlation of the 'Human-baseline' [Malinowski and Fritz 2014a] (at y-axis).

10.6 Additional Material

Here, we provide qualitative examples of different variants of our architecture and show the correlations of predicted answer words and question words with human answer and question words. The examples are chosen to highlight challenges as well as differences between 'Ask Your Neurons' and 'Language only' architectures. Table 10.17 also shows a few failure cases. In all cases but 'multiple words answer', we use the best 'single word' variants. Although 'Language only' ignores the image, it is still able to make 'reasonable guesses' by exploiting biases captured by the dataset that can be viewed as a type of common sense knowledge. For instance, 'tea kettle' often sits on the oven, cabinets are usually 'brown', 'chair' is typically placed in front of a table, and we commonly keep a 'photo' on a cabinet (Table 10.10, 10.12, 10.13, 10.16). This effect is analysed in Figure 10.6. Each data point in the plot represents the correlation between a question and a predicted answer words for our 'Language only' model (x-axis) versus the correlation in the human answers (y-axis).

Despite the reasonable guesses of the 'Language only' architecture, the 'Ask Your Neurons' predicts in average better answers (Table 10.1 that is replicated for the convenience of the reader) by exploiting the visual content of images. For instance in Table 10.14 the 'Language only' model incorrectly answers '6' on the question 'How many burner knobs are there ?' because it has seen only this answer during the training with exactly the same question but on different image.

Both models, 'Language only' and 'Ask Your Neurons', have difficulties to answer correctly on long questions or such questions that expect a larger number of answer words (Table 10.17). On the other hand both models are doing well on predicting a type of the question (e.g. 'what color ...' result in a color name in the answer, or 'how many ...' questions result in a number), there are a few rare cases with an incorrect type of the predicted answer (the last example in Table 10.17).

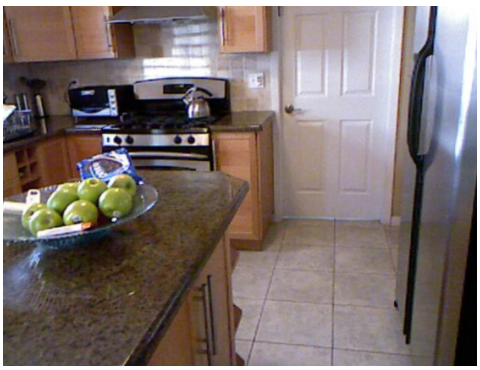
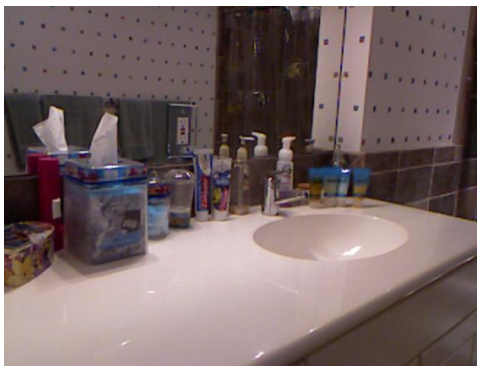
	 What are the objects close to the wall?	 What is on the stove?	 What is left of sink?
Ask Your Neurons:	wall decoration	tea kettle	tissue roll
Language only:	books	tea kettle	towel
Ground truth answers:	wall decoration	tea kettle	tissue roll

Table 10.10: Examples of compound answer words.

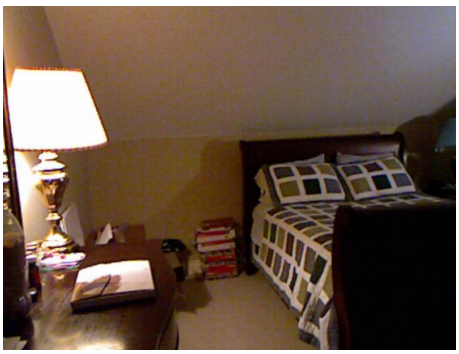
	 How many lamps are there?	 How many pillows are there on the bed?	 How many pillows are there on the sofa?
Ask Your Neurons:	2	2	3
Language only:	2	3	3
Ground truth answers:	2	2	3

Table 10.11: Counting questions.

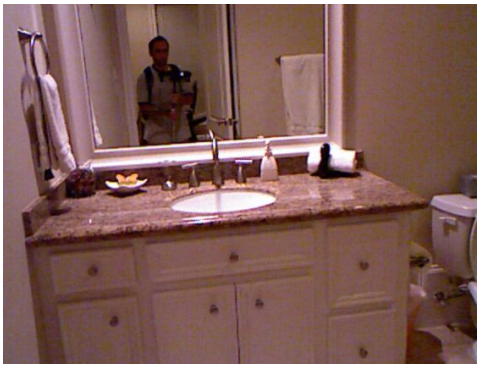
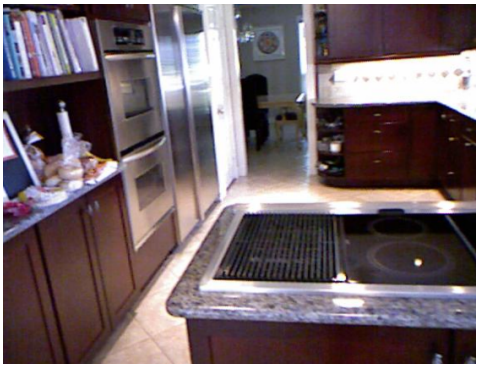
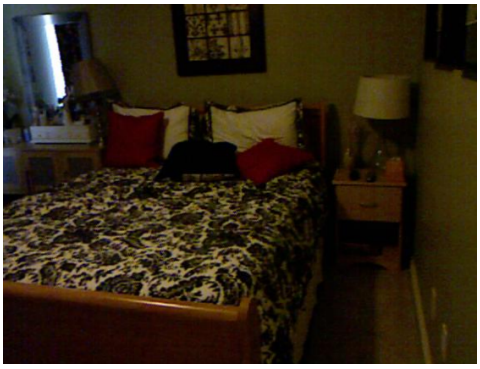
	 What color is the towel?	 What color are the cabinets?	 What is the colour of the pillows?
Ask Your Neurons:	brown	brown	black, white
Language only:	white	brown	blue, green, red
Ground truth answers:	white	brown	black, red, white

Table 10.12: Questions about color.

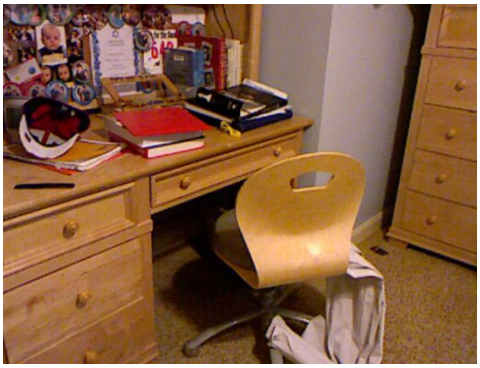
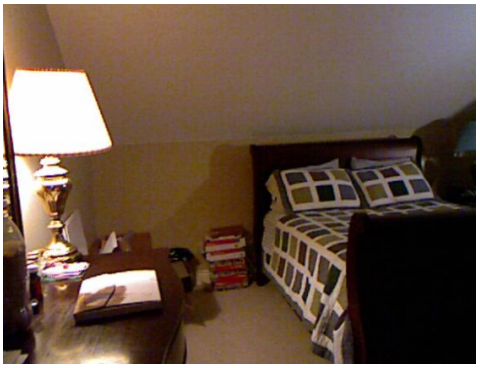
	 What is hanged on the chair?	 What is the object close to the sink?	 What is the object on the table in the corner?
Ask Your Neurons:	clothes	faucet	lamp
Language only:	jacket	faucet	plant
Ground truth answers:	clothes	faucet	lamp

Table 10.13: Correct answers by our 'Ask Your Neurons' architecture.

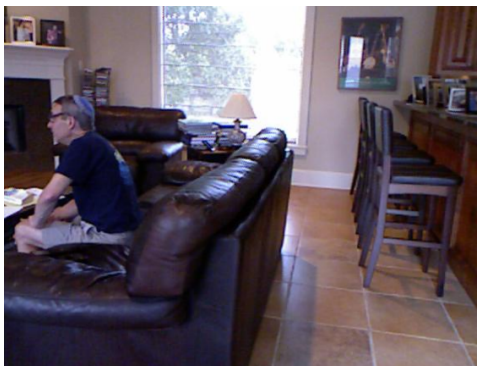
	 What are the things on the cabinet?	 What is in front of the shelf?	 How many burner knobs are there?
Ask Your Neurons:	photo	chair	4
Language only:	photo	basket	6
Ground truth answers:	photo	chair	4

Table 10.14: Correct answers by our 'Ask Your Neurons' architecture.

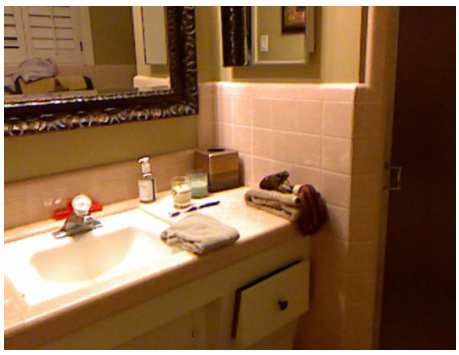
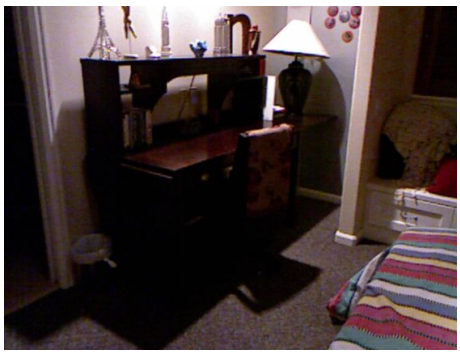
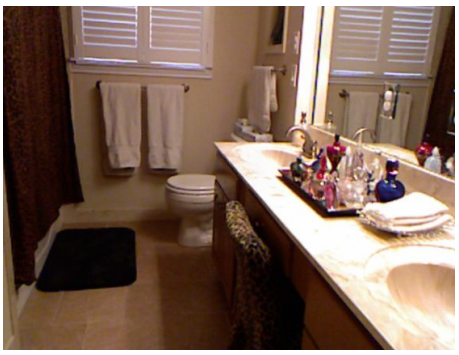
	 What is the object close to the counter?	 What is the colour of the table and chair?	 How many towels are hanged?
Ask Your Neurons:	sink	brown	3
Language only:	stove	brown	4
Ground truth answers:	sink	brown	3

Table 10.15: Correct answers by our 'Ask Your Neurons' architecture.

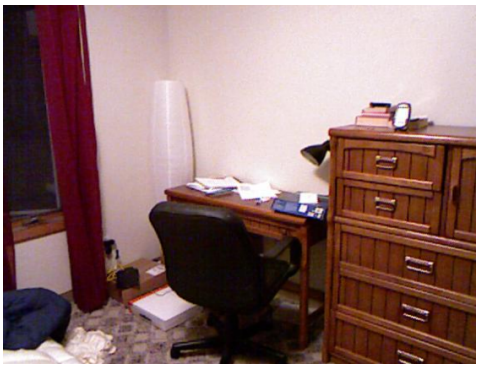
			
What is on the right most side on the table?		What are the things on the coffee table?	What is in front of the table?
Ask Your Neurons:	lamp	books	chair
Language only:	machine	jacket	chair
Ground truth answers:	lamp	books	chair

Table 10.16: Correct answers by our 'Ask Your Neurons' architecture.

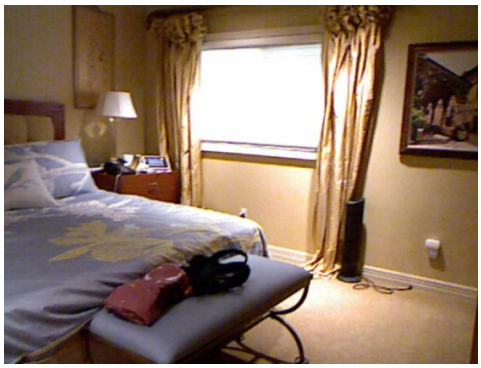
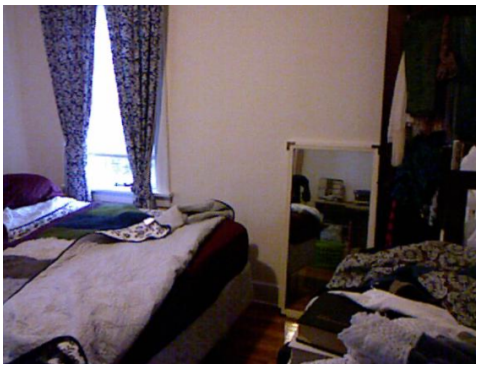
			
What is on the left side of the white oven on the floor and on right side of the blue armchair?		What are the things on the cabinet?	What color is the frame of the mirror close to the wardrobe?
Ask Your Neurons:	oven	chair, lamp, photo	pink
Language only:	exercise equipment	candelabra	curtain
Ground truth answers:	garbage bin	lamp, photo, telephone	white

Table 10.17: Failure cases.

Chapter 11

Ask Your Neurons: A Deeper Analysis

Contents

Since our proposed Visual Turing Test presented in Chapter 8 and Chapter 9, we have witnessed an increased interests of the research community on this task resulting in many novel methods as well as new datasets. Most notably, Antol et al. [2015] have proposed a large-scale variant of a Visual Turing Test named "Visual Question Answering", or shortly VQA. The advent of such large volume datasets serves as an excellent testbed for different data-driven approaches such as the one presented in Chapter 10. It turns out that many neural-based approaches can be framed as special cases of a more general encoder-decoder framework for a Visual Turing Test that considers four different 'modules': visual and question encoders, an answer decoder, and a multimodal embedding. This new perspective encapsulates advances in particular fields such as machine recognition or language understanding, and allows to analyse an impact of different design choices on the overall performance on the task. We believe that further generations of holistic machines should also follow a perspective of this sort.

11.1 Introduction

With the recent advances in natural language and image understanding, more complex, demanding and holistic tasks become within our reach. Such advances have contributed in the development of very deep recognition architectures that achieve near human performance on the ImageNet dataset [He et al. 2015]. At the same time we also observe alternative to LSTM [Hochreiter and Schmidhuber 1997] or Bag-Of-Words [Manning et al. 2008] neural-based approaches to model natural language. For instance, Convolutional Neural Networks can be naturally extended to work with natural language [Kim 2014; Kalchbrenner et al. 2014]. The latter approach is especially appealing as it sparkles a hope for "one learnable model". In practice, such models may, arguably, also benefit from advances in object recognition.

In contrast to the work of Malinowski et al. [2015] who have presented a monolithic question answering about images architecture, we take an alternative encoder-decoder perspective and decompose the method of Malinowski et al. [2015] into several 'modules'. Most important, each module can be replaced and its influence to other parts as well as to the whole task can be readily studied. Concretely, we decompose the visual question answering architectures into the following modules: visual encoder, question encoder, answer decoder, and multimodal embedding. An abstract depiction of the proposed view is presented in Figure 11.1. We also frame the work of Malinowski et al. [2015] that combines LSTM with CNN via a multimodal concatenation as a special case of the newly proposed framework for a Visual Turing Test. Moreover, such 'modular' perspective allows us to also study different design choices on a large scale visual question answering dataset VQA [Antol et al. 2015], and lead to a better model. Our analysis shows that a stronger visual component and multimodal embedding are crucial in achieving better results. The lessons learnt on VQA are also transferable to DAQUAR [Malinowski and Fritz 2014a] leading to a competitive model that yet uses a global, full-frame image representation.

11.2 Related Work

Since we have proposed a modern approach to a Visual Turing Test [Malinowski and Fritz 2014a,b, 2015], frequently also referred to as "Visual Question Answering", there has been a strong interest in this task. In the following we first discuss related tasks and subtasks, then approaches to tackle the Visual Turing Test and datasets proposed for it. Finally, we discuss the relations to our work.

11.2.1 Convolutional neural networks for visual recognition

One component to answer questions about images is to extract information from visual content. Since the proposal of AlexNet [Krizhevsky et al. 2012], Convolutional Neural Networks (CNNs) have become dominant and most successful approaches to extract relevant representation from the image. CNNs directly learn the representation from the raw image data and are trained on large image corpora, typically ImageNet [Russakovsky et al. 2014]. Interestingly, after these models are pre-trained on ImageNet, they can typically be adapted for other tasks. In this work, we evaluate how well the most dominant and successful

CNN models can be adapted for the Visual Turing Task. Specifically, we evaluate *AlexNet* [Krizhevsky et al. 2012], *VGG* [Simonyan and Zisserman 2015], *GoogleNet* [Szegedy et al. 2015], and *ResNet* [He et al. 2015]. These models, reportedly, achieve increasingly better accuracies on the ImageNet dataset, and hence, arguably, serve as stronger models of visual perception.

11.2.2 Encodings for text sequence understanding

The other important component to answer a question about an image is to understand the natural language question, which means here building a representation of a variable length sequence of words (or characters, but we will focus only on the words in this work). The first approach is to encode all words of the question as a Bag-Of-Words [Manning and Schütze 1999], and hence ignoring an order in the sequence of words. Another option is to use, similar to the image encoding, a CNN with pooling to handle variable length input [Kim 2014; Kalchbrenner et al. 2014]. Finally, Recurrent Neural Networks (RNNs) are methods developed to directly handle sequences, and have shown recent success on natural language tasks such as machine translation [Cho et al. 2014; Sutskever et al. 2014]. In this work we investigate a Bag-Of-Words (BOW), a CNN, and two RNN variants (LSTM [Hochreiter and Schmidhuber 1997] and GRU [Cho et al. 2014]) to encode the question.

11.2.3 Combining RNNs and CNNs for description of visual content.

The task of describing visual content like still images as well as videos has been successfully addressed with a combination of encoding the image with CNNs and decoding, i.e. predicting the sentence description with an RNN [Donahue et al. 2015; Karpathy and Fei-Fei 2015; Venugopalan et al. 2015b; Vinyals et al. 2014; Zitnick et al. 2013]. This is achieved by using the RNN model that first gets to observe the visual content and is trained to afterwards predict a sequence of words that is a description of the visual content. Our work extends this idea to question answering, where we formulate a model trained to either generate or classify an answer based on visual as well as natural language input.

11.2.4 Grounding of natural language and visual concepts.

Dealing with natural language input does involve the association of words with meaning. This is often referred to as the grounding problem - in particular if the "meaning" is associated with a sensory input. While such problems have been historically addressed by symbolic semantic parsing techniques [Krishnamurthy and Kollar 2013; Matuszek et al. 2012], there is a recent trend of machine learning-based approaches [Karpathy and Fei-Fei 2015; Karpathy et al. 2014; Akata et al. 2016; Kong et al. 2014; Hu et al. 2016; Rohrbach et al. 2015a; Mao et al. 2016] to find the associations. Answering questions about images can be interpreted as first grounding the question in the image and then predicting an answer. Our approach thus is similar to the latter approaches in that we do not enforce or evaluate any particular representation of "meaning" on the language or image modality. We treat this as latent and

leave it to the joint training approach to establish an appropriate hidden representation to link the visual and textual representations.

11.2.5 Textual question answering.

Answering on purely textual questions has been studied in the NLP community [Berant and Liang 2014; Liang et al. 2013] and state of the art techniques typically employ semantic parsing to arrive at a logical form capturing the intended meaning and infer relevant answers. Only recently, the success of the previously mentioned neural sequence models, namely RNNs, has carried over to this task [Iyyer et al. 2014; Weston et al. 2014]. More specifically Iyyer et al. [2014] use dependency-tree Recursive NN instead of LSTM, and reduce the question-answering problem to a classification task. Weston et al. [2014] propose different kind of network - memory networks - that is used to answer questions about short stories. In their work, all the parts of the story are embedded into different "memory cells", and next a network is trained to attend to relevant cells based on the question and decode an answer from that. A similar idea has also been applied to question answering about images, for instance by Yang et al. [2015].

11.2.6 Visual Turing Test

Recently, a large number architectures have been proposed to approach the Visual Turing Test [Malinowski and Fritz 2014b], frequently also referred to as "Visual Question Answering". They range from symbolic to neural based approaches. There are also architectures that combine both symbolic and neural paradigms together. Some approaches use explicit visual representation in the form of bounding boxes surrounding objects of interest, while other use global full frame image representation, or soft attention mechanism. Yet others use an external knowledge base that helps in answering questions.

Symbolic based approaches. In our first work on Visual Turing Test [Malinowski and Fritz 2014a], we present a question answering system based on a semantic parser on a varied set of human question-answer pairs. Although it is the first attempt to handle question answering on DAQUAR, and despite its introspective benefits, it is a rule-based approach that requires a careful schema crafting, is not that scalable, and finally it strongly depends on the output of visual analysis methods as joint training in this model is not yet possible. Due to such limitations, the community has rather shifted towards either neural based or combined approaches.

Deep Neural Approaches with full frame CNN. Most contemporary approaches use a global image representation, i.e. they encode the whole image with a CNN. Questions are then encoded with an RNN [Malinowski et al. 2015; Ren et al. 2015a; Gao et al. 2015] or a CNN [Ma et al. 2015]. In contrast to symbolic based approaches, neural based architectures offer scalable and joint end-to-end training that liberates them from ontological commitment that would otherwise be introduced by a semantic parser. Moreover, such approaches are not 'hard' conditioned on the visual input and therefore can naturally take advantage of different

language biases in question answer pairs, which can be interpret as learning common sense knowledge.

Attention-based Approaches. Following Xu et al. [2015], who proposed to use spatial attention for image description, Yang et al. [2015]; Xu and Saenko [2015]; Zhu et al. [2016]; Chen et al. [2015]; Shih et al. [2016] predict a latent weighting (attention) of spatially localized images features (typically a convolutional layer of the CNN) based on the question. The weighted image representation rather than the full frame feature representation is then used as a basis for answering the question. In contrast to the previous models using attention, Dynamic Memory Networks (DMN) [Kumar et al. 2016; Xiong et al. 2016] first pass all spatial image features through a bi-directional GRU that captures spatial information from the neighboring image patches, and next retrieve an answer from a recurrent attention based neural network that allows to focus only on a subset of the visual features extracted in the first pass. Another interesting direction has been taken by Ilievski et al. [2016] who run state-of-the-art object detector of the classes extracted from the key words in the question. In contrast to other attention mechanisms, such approach offers a focused, question dependent, "hard" attention.

Answering with an external knowledge base. Wu et al. [2016b] argue for an approach that first represents an image as an intermediate semantic attribute representation, and next query external knowledge sources based on the most prominent attributes and relate them to the question. With the help of such external knowledge base, such approach captures richer semantic representation of the world, beyond what is directly contained in images.

Compositional approaches. A different direction is taken by Andreas et al. [2016b] who predict the most important components to answer the question with a natural language parser. The components are then mapped to neural modules, which are composed to a deep neural network based on the parse tree. While each question induces a different network, the modules are trained jointly across questions. This work compares to Malinowski and Fritz [2014a] by exploiting explicit assumptions about the compositionality of natural language sentences. Related to the Visual Turing Test, Malinowski and Fritz [2014c] have also combined a neural based representation with the compositionality of the language for the text-to-image retrieval task.

Dynamic parameters. Noh et al. [2015b] have an image recognition network and a Recurrent Neural Network (GRU) that dynamically change the parameters (weights) of visual representation based on the question. More precisely, the parameters of its second last layer are dynamically predicted from the question encoder network and in this way changing for each question. While question encoding and image encoding is pre-trained, the network learns parameter prediction only from image-question-answer triples.

11.2.7 Datasets for visual question answering

Datasets are a driving force for the recent progress in visual question answering. A large number of visual question answering datasets have recently been proposed. The first proposed datasets is DAQUAR [Malinowski and Fritz 2014a], which contains about 12.5 thousands manually annotated question-answer pairs about 1449 indoor scenes [Silberman et al. 2012]. While the dataset has originally contained a single answer (that can consist of multiple words) per question, in this work we extend the dataset by collecting additional answers for each questions. This captures uncertainties in evaluation. We evaluate our approach on this dataset and discuss several consensus evaluation metrics that take the extended annotations into account. In parallel to our Visual Turing Test, Geman et al. [2015] developed another Visual Turing Test. Their work, however, focuses on yes/no type of questions, and provide detailed object-scene annotations.

Shortly after the introduction of DAQUAR, three other large-scale datasets have been proposed. All are based on MS-COCO [Lin et al. 2014b]. Gao et al. [2015] have annotated about $158k$ images with $316k$ Chinese question answer pairs together with their corresponding English translations. Ren et al. [2015a] have taken advantage of the existing annotations for the purpose of image description generation task and transform them into question answer pairs with the help of a set of hand-designed rules and a syntactic parser [Klein and Manning 2003]. This procedure has approximately generated $118k$ question answer pairs. Finally, arguably nowadays the most popular, large scale dataset on question answering about images is VQA [Antol et al. 2015]. It has approximately $614k$ questions about the visual content of about $205k$ real-world images. Similarly to our Consensus idea, VQA provides 10 answers per each image. For the purpose of the challenge the test answers are not publicly available. We perform one part of the experimental analysis in this paper on the VQA dataset, examining different variants of our proposed approach.

Although simple, automatic performance evaluation metrics have been a part of building first visual question answering datasets [Malinowski and Fritz 2014a,b, 2015], Yu et al. [2015b] have simplified the evaluation even further by introducing Visual Madlibs - a multiple choice question answering by filling the blanks task. In this task, a question answering architecture has to choose one out of four provided answers for a given image and the prompt. Formulating question answering task in this way has wiped out ambiguities in answers, and just a simple accuracy metric can be used to evaluate different architectures on this task. Yet, the task requires holistic reasoning about the images, and despite of simple evaluation, it remains challenging for machines.

The Visual7W [Zhu et al. 2016] extends canonical question and answer pairs with additional groundings of all objects appearing in the questions and answers to the image by annotating the correspondences. It contains natural language answers, but also answers which require to locate the object, which is then similar to the task of explicit grounding discussed above. Visual7W builds question answer pairs based on the Visual Genome dataset [Krishna et al. 2016], and contains about $330k$ questions. In contrast to others such as VQA [Antol et al. 2015] or DAQUAR [Malinowski and Fritz 2014a] that has collected unconstrained question answer pairs, the Visual Genome focuses on the six, so called, Ws: what, where, when, who, why, and how, which can be answered with a natural language answer. An

additional 7th question – which – requires a bounding box location as answer. Similarly to Visual Madlibs [Yu et al. 2015b], Visual7W also contains multiple-choice answers.

Related to Visual Turing Test, Chowdhury et al. [2016a] have proposed collective memories and Xplore-M-Ego - a dataset of images with natural language queries, and a media retrieval system. This work focuses on a user centric, dynamic scenario, where the provided answers are conditioned not only on questions but also on the geographical position of the questioner.

Moving from asking questions about images to questions about video enhances typical questions with temporal structure. Zhu et al. [2015] propose a task which requires to fill in blanks the captions associated with videos. The task requires inferring the past, describing the present and predicting the future in a diverse set of video description data ranging from cooking videos [Regneri et al. 2013] over web videos [Trecvid 2014] to movies [Rohrbach et al. 2015b]. Tapaswi et al. [2016] propose MovieQA, which requires to understand long term connections in the plot of the movie. Given the difficulty of the data, both works provide multiple-choice answers.

11.2.8 Relations to our work.

The original version of this work [Malinowski et al. 2015] belongs to the category of "Deep Neural Approaches with full frame CNN", and is among the very first methods of this kind (Section 10.3). We extend [Malinowski et al. 2015] by introducing a more general and modular encoder-decoder perspective (Section 11.2.9) that encapsulates a few different neural approaches. Next, we broaden our original analysis done on DAQUAR (Section 10.4) to the analysis of different neural based approaches on VQA showing the importance of getting a few details right together with benefits of a stronger visual encoder (Section 11.3). Finally, we transfer lessons learnt from VQA [Antol et al. 2015] to DAQUAR [Malinowski and Fritz 2014a], showing a significant improvement on this challenging task (Section 11.3).

11.2.9 Encoder-decoder Perspective on Visual Turing Test

In the previous Section 10.3 we have described a way to model visual question answering with a single recurrent network for question and image encoding and answering, in this section we describe a modular framework where a question encoder has to be combined with a visual encoder in order to produce answers with an answer decoder (Figure 11.1). This conceptually modular representation is helpful in investigating the behavior of the whole architecture while different encoders, multimodal embeddings, and decoders are used.

11.2.9.1 Question encoders

The main goal of a question encoder is to capture a meaning of the question, which we write here as $\Psi(\boldsymbol{q})$. Such an encoder can range from a very structured one like Semantic Parser used in Malinowski and Fritz [2014a] and Liang et al. [2013] that explicitly model compositional nature of the question, to structureless Bag-Of-Word (BOW) approaches that temporarily sum up the input question words (Figure 11.3). In this work, we investigate a few encoders within such a spectrum. Two recurrent question encoders under our investigation,

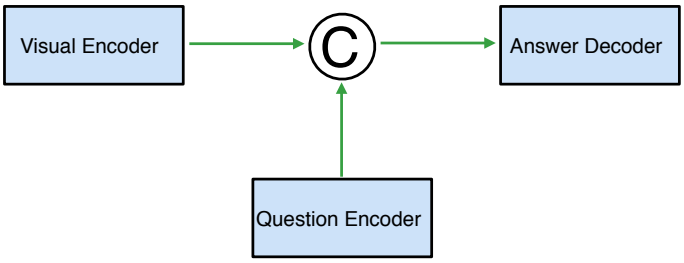


Figure 11.1: Our *Refined Ask Your Neurons* architecture for answering questions about images that includes the following modules: visual and question encoders, and answer decoder. A multimodal embedding C combines both encodings into a joint space that the decoder decodes from. See Section 11.2.9 for details.

that is LSTM [Hochreiter and Schmidhuber 1997] (see Section 10.3) and GRU [Cho et al. 2014], assume a temporal ordering in questions. Moreover, we also investigate an orderless, and already aforementioned BOW.

Long-Short Term Memory (LSTM). LSTM is a recurrent neural network that models a temporal dynamics by encoding an input sequence into its hidden states. Every recurrent unit depends on the input variable as well as the previous state. To deal with the 'vanishing gradient' problem, LSTM uses different gates that controls the flow of information. LSTM is expressed by the following set of equations:

$$\boldsymbol{i}_t = \sigma(W_{vi}\boldsymbol{v}_t + W_{hi}\boldsymbol{h}_{t-1} + \boldsymbol{b}_i) \tag{11.1}$$
$$\boldsymbol{f}_t = \sigma(W_{vf}\boldsymbol{v}_t + W_{hf}\boldsymbol{h}_{t-1} + \boldsymbol{b}_f) \tag{11.2}$$
$$\boldsymbol{o}_t = \sigma(W_{vo}\boldsymbol{v}_t + W_{ho}\boldsymbol{h}_{t-1} + \boldsymbol{b}_o) \tag{11.3}$$
$$\boldsymbol{g}_t = \phi(W_{vg}\boldsymbol{v}_t + W_{hg}\boldsymbol{h}_{t-1} + \boldsymbol{b}_g) \tag{11.4}$$
$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \boldsymbol{g}_t \tag{11.5}$$
$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \phi(\boldsymbol{c}_t) \tag{11.6}$$

where σ is the sigmoid function, ϕ is the hyperbolic tangent, and $\boldsymbol{v}_t$, $\boldsymbol{h}_t$ are input and hidden state at time t. Variable $\boldsymbol{c}_t$ represents memory cells.

Gated Recurrent Unit (GRU). GRU is a simpler variant of LSTM that also uses gates (a reset gate $\boldsymbol{r}$ and an update gate $\boldsymbol{u}$) in order to keep long term dependencies. GRU is expressed by the following set of equations:

$$\boldsymbol{r}_t = \sigma(W_{vr}\boldsymbol{v}_t + W_{hr}\boldsymbol{h}_{t-1} + \boldsymbol{b}_r) \tag{11.7}$$
$$\boldsymbol{u}_t = \sigma(W_{vu}\boldsymbol{v}_t + W_{hu}\boldsymbol{h}_{t-1} + \boldsymbol{b}_u) \tag{11.8}$$
$$\boldsymbol{c}_t = W_{vc}\boldsymbol{v}_t + W_{hc}(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}) + \boldsymbol{b}_c \tag{11.9}$$
$$\boldsymbol{h}_t = \boldsymbol{u}_t \odot \boldsymbol{h}_{t-1} + (\mathbf{1} - \boldsymbol{u}_t) \odot \phi(\boldsymbol{c}_t) \tag{11.10}$$

where σ is the sigmoid function, ϕ is the hyperbolic tangent, and $\boldsymbol{v}_t$, $\boldsymbol{h}_t$ are input and hidden state at time t. The representation of the question $\boldsymbol{q}$ is the hidden vector at last time step,

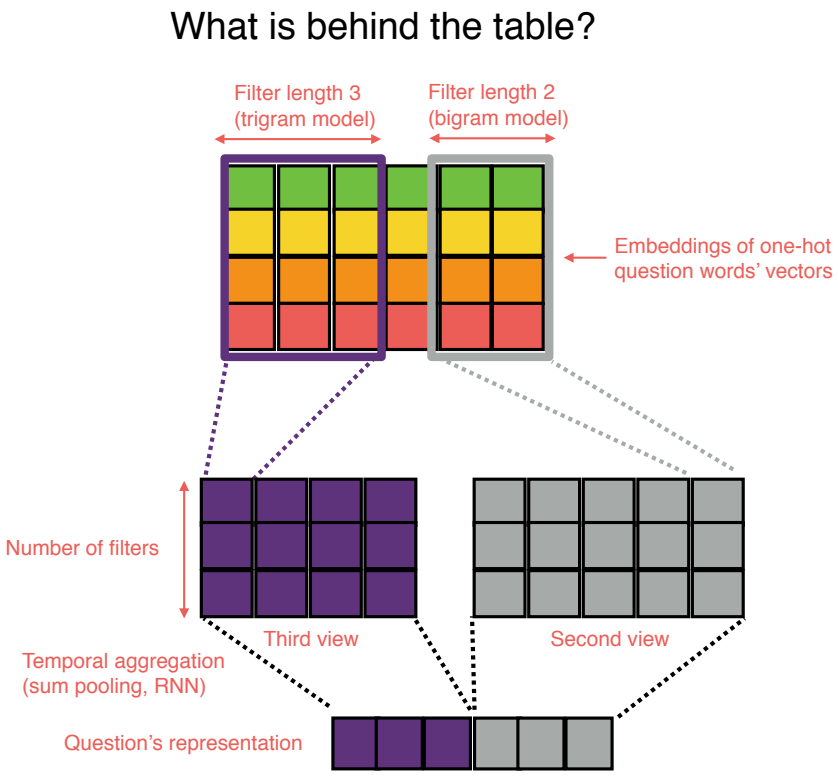


Figure 11.2: CNN for encoding the question that convolves word embeddings (learnt or pre-trained) with different kernels, second and third views are shown, see Section 11.2.9.1 and Yang et al. [2015] for details.

i.e. $\Psi_{\text{RNN}}(\boldsymbol{q}) := \boldsymbol{h}_T$.

Bag-Of-Word (BOW). Conceptually the simplest, the BOW approach (Figure 11.3) sums up over the words embeddings:

$$\Psi_{\text{BOW}}(\boldsymbol{q}) := \sum_{t}^{n} W_e(\boldsymbol{q}_t). \tag{11.11}$$

where $\boldsymbol{W}_e$ is a matrix and $\boldsymbol{q}_t$ is one-hot binary vector of the word with exactly one 1 pointing to a place of the 'word' in the vocabulary (Figure 11.3). BOW rejects words ordering in the question, so that especially questions with swapped arguments of spatial prepositions become indistinguishable, i.e.
Ψ_{BOW}(red chair left of sofa) = Ψ_{BOW}(red sofa left of chair) in the BOW sentence representation.

Convolutional Neural Network (CNN). Convolutional Neural Network (CNN) that models language [Kim 2014; Kalchbrenner et al. 2014; Ma et al. 2015; Yang et al. 2015] is gaining popularity due to its speed and good accuracy for the language-oriented tasks. Since it considers a larger context, it arguably maintains more structure than BOW but does not

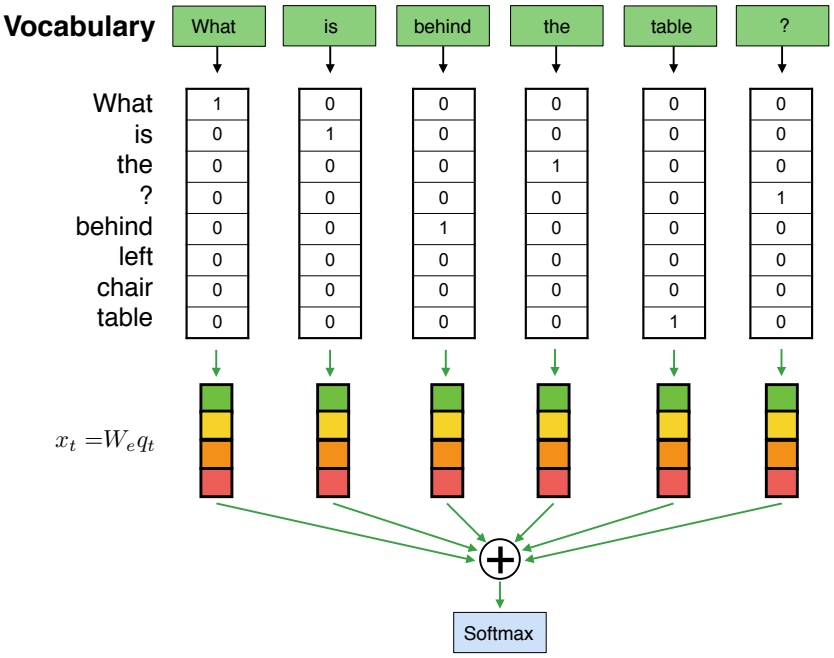


Figure 11.3: Bag-Of-Words (BOW) for encoding the question, see Section 11.2.9.1 for details.

model such long term dependencies as recurrent neural networks. Figure 11.2 depicts our CNN architecture, which is very similar to Ma et al. [2015] and Yang et al. [2015], that convolves word embeddings (we either learn it jointly with the whole model or use GLOVE [Pennington et al. 2014] in our experiments) with three convolutional kernels of length 1, 2 and 3 (for the sake of clarity, we only show two kernels in the Figure). We call such architecture with 1, ..., n kernel lengths n views CNN. At the end, the kernel's outputs are temporarily aggregated for the final question's representation. We use either sum pooling or a recurrent neural network (CNN-RNN) to accomplish this step.

11.2.9.2 Visual encoders

The second important component of the encoder-decoder architectures for Visual Turing Test is visual representation. Nowadays, Convolutional Neural Networks (CNNs) become the state-of-the-art framework that provide features from images. The typical protocol of using the visual models is to first pre-train them on the ImageNet dataset [Russakovsky et al. 2014], a large scale recognition dataset, and next use them as an input for the rest of the architecture. Fine-tuning the weights of the encoder to the task at hand is also possible. In our experiments, we use chronologically the oldest CNN architecture fully trained on ImageNet – a Caffe implementation of AlexNet [Jia et al. 2014; Krizhevsky et al. 2012] – as well as the recently introduced deeper networks – Caffe implementations of GoogLeNet and VGG [Szegedy et al. 2015; Simonyan and Zisserman 2015] – to the most recent extremely deep architectures – a Facebook implementation of 152 layered ResidualNet [He et al. 2015]. As can be seen from our experiments in Section 11.3, a strong visual encoder plays an important role in Visual Turing Test.

11.2.9.3 Multimodal embedding

The presented neural question encoders transform linguistic question into a vector space. Similarly visual encoders encode images as vectors. A multimodal fusion module combines both vector spaces into another vector space that decoding of answers is feasible. Let $\Psi(\boldsymbol{q})$ be a question representation (BOW, CNN, LSTM, GRU), and $\Phi(\boldsymbol{x})$ be a representation of an image. Then $C(\Psi(\boldsymbol{q}), \Phi(\boldsymbol{x}))$ is a function which embeds both vectors. In this work, we investigate three multimodal embedding techniques: Concatenation, piecewise multiplication, and summation. Since the last two techniques require compatibility in the number of feature components, we use additional visual embedding matrix $W_{ve} \in \mathbb{R}^{|\Psi(\boldsymbol{q})| \times |\Phi(\boldsymbol{x})|}$. Let W be weights of an answer decoder. Then we have $WC(\Psi(\boldsymbol{q}), \Phi(\boldsymbol{x}))$, which is

$$W_q\Psi(\boldsymbol{q}) + W_v\Phi(\boldsymbol{x}) \tag{11.12}$$

$$W(\Psi(\boldsymbol{q}) \odot W_{ve}\Phi(\boldsymbol{x})) \tag{11.13}$$

$$W\Psi(\boldsymbol{q}) + WW_{ve}\Phi(\boldsymbol{x}) \tag{11.14}$$

in concatenation, piecewise multiplication, and summation fusion techniques respectively. In Equation 11.12, we decompose W into two matrices W_q and W_v, that is $W = [W_q; W_v]$. In Equation 11.13, $\odot$ is a piecewise multiplication. Similarity between Equation 11.12 and Equation 11.14 is interesting as the latter is the former with weight sharing and additional decomposition into WW_{ve}.

11.2.9.4 Answer decoders

Answer words generation. The last component of the encoder-decoder architecture for Visual Turing Test (Figure 11.1) is an answer decoder. Malinowski et al. [2015], inspired by the work on the image description task [Donahue et al. 2015], uses an LSTM as decoder that shares the parameters with the encoder.

Classification. An alternative approach that cast answering problem as a classification task, with answers as different classes, has recently gained popularity, especially in VQA task [Antol et al. 2015]. Thorough this work, we investigate both approaches.

11.3 Analysis on VQA

While Section 10.4 analyses our original architecture [Malinowski et al. 2015] on the DAQUAR dataset, in this section, we analyze different variants and design choices for neural question answering on the large-scale Visual Question Answering (VQA) dataset [Antol et al. 2015]. It is currently one of the largest and most popular visual question answering dataset with human question answer pairs. In the following, after describing the experimental setup (Section 11.3.1), we first describe several experiments which examine the different variants of question encoding, only looking at language input to predict the answer (Section 11.3.1), and then, we examine the full model (Section 11.3.3).

kernel length k	single view $= k$	multi view $\leq k$
1	47.43	47.43
2	48.11	48.06
3	**48.26**	48.09
4	**48.27**	47.86

Table 11.1: Results on VQA validation set, "Question-only" model: Analysis of CNN questions encoders with different filter lengths, accuracy in %, see Section 11.3.2.1 for discussion.

11.3.1 Experimental setup

We evaluate on the VQA dataset [Antol et al. 2015], which is built on top of the MS-COCO dataset [Lin et al. 2014b]. Although VQA offers a different challenge tasks, we focus our efforts on the Real Open-Ended Visual Question Answering challenge. The challenge consists of 10 answers per question with about $248k$ training questions, about $122k$ validation questions, and about $244k$ test questions.

As VQA consist mostly of single word answers (over 89%), we treat the question answering problem as a classification problem of the most frequent answers in the training set. For the evaluation of the different model variants and design choices, we train on the training set and test on the validation set. Only the final evaluations (Table 11.8) are evaluated on the test set of the VQA challenge, we evaluate on both parts test-dev and test-standard, where for the latter the answers are not publicly available. As a performance measure we use a Consensus variant of Accuracy introduced in Antol et al. [2015], where the predicted answer gets score between 0 and 1, with 1 if it matches with at least three human answers. We use ADAM [Kingma and Ba 2014] throughout our experiments as we found out it performs better than SGD with momentum. We keep default hyper-parameters for ADAM. Employed Recurrent Neural Networks maps input question into 500 dimensional vector representation. All the CNNs for text are using 500 feature maps in our experiments, but the output dimensionality also depends on the number of views. In preliminary experiments we found that removing question mark '?' in the questions slightly improves the results, and we report the numbers only with this setting. Since VQA has 10 answers associated with each question, we need to consider a suitable training strategy that takes this into account. We have examined the following strategies: picking an answer randomly, randomly but if possible annotated as confidently answered, all answers, or choosing the most frequent answer. In the following, we only report the results using the last strategy as we have found out little difference in accuracy between the strategies. To allow training and evaluating many different models with limited time and computational power, we do not fine-tune the visual representations in these experiments, although our model would allow us to do so. All the models, which are publicly available under `https://github.com/mateuszmalinowski/Kraino`, are implemented in Keras [Chollet 2015] and Theano [Bastien et al. 2012].

Question encoder	Word embedding learned	GLOVE
BOW	47.41	47.91
CNN	48.26	48.53
GRU	47.60	48.11
LSTM	47.80	**48.58**

Table 11.2: Results on VQA validation set, "Question-only" model: Analysis of different questions encoders, accuracy in %, see Section 11.3.2 for discussion.

	top frequent answers		
Encoder	1000	2000	3000
BOW	47.91	48.13	47.94
CNN	48.53	48.67	48.57
LSTM	48.58	**48.86**	48.65

Table 11.3: Results on VQA validation set, "Question-only" model: Analysis of the number of top frequent answer classes, with different question encoders. All using GLOVE; accuracy in %; see Section 11.3.2.4 for discussion.

11.3.2 Question-only

We start our analysis from "Question-only" models that do not use images to answer on questions. Note that the "Question-only" baselines play an important role in the question answering about images tasks since it clearly studies effects of added vision. Hence, better overall performance of the model is not obscured by a better language model. To understand better different design choices, we have conducted our analysis along the different 'design' dimensions.

11.3.2.1 CNN questions encoder

We first examine different hyper-parameters for CNNs to encode the question. We first consider the filter's length of the convolutional kernel. We run the model over different kernel lengths ranging from 1 to 4 (Table 11.1, left column). We notice that increasing the kernel lengths improves performance up to length 3 were the performance levels out, we thus use kernel length 3 in the following experiments for, such CNN can be interpreted as a trigram model. We also tried to run simultaneously a few kernels with different lengths. In Table 11.1 (right column) one view corresponds to a kernel length 1, two views correspond to two kernels with length 1 and 2, three views correspond to length 1, 2 and 3, etc. However, we find that the best performance still achieve with a single view and kernel length 3 or 4.

	no norm	L2 norm
Concatenation	47.21	52.39
Summation	40.67	**53.27**
Piece-wise multiplication	49.50	52.70

Table 11.4: Results on VQA validation set, vision and language: Analysis of different multimodal techniques that combine vision with language on BOW (with GLOVE word embedding and VGG-19 fc7), accuracy in %, see Section 11.3.3.1.

11.3.2.2 BOW questions encoder

Alternatively to neural network encoders, we consider Bag-Of-Words (BOW) approach where one-hot representations of the question words are first mapped to a shared embedding space, and subsequently summed over (Equation 11.11), i.e. $\Psi(\text{question}) := \sum_{\text{word}} W_e(word)$. Surprisingly, such a simple approach gives very competitive results (first row in Table 11.2) compared to the CNN encoding discussed in the previous section (second row).

Recurrent questions encoder We examine two recurrent questions encoders, LSTM [Hochreiter and Schmidhuber 1997] and a simpler GRU [Cho et al. 2014]. The last two rows of Table 11.2 show a slight advantage of using LSTM.

11.3.2.3 Pre-trained words embedding

In all the previous experiments, we jointly learn the embedding transformation $\boldsymbol{W}_e$ together with the whole architecture only on the VQA dataset. This means we do not have any means for dealing with unknown words in questions at test time apart from using a special token $\langle$UNK$\rangle$ to indicate such class. To address such shortcoming, we investigate the pre-trained word embedding transformation GLOVE [Pennington et al. 2014] that encodes question words (technically it maps one-hot vector into a 300 dimensional real vector). This choice naturally extends the vocabulary of the question words to about 2 million words extracted a large corpus of web data – Common Crawl [Pennington et al. 2014] – that is used to train the GLOVE embedding. Since the BOW architecture in this scenario becomes shallow (only classification weights are learnt), we add an extra hidden layer between pooling and classification (without this embedding, accuracy drops by 5%). Table 11.2 (right column) summarizes our experiments with GLOVE. For all question encoders, the word embedding consistently improves performance which confirms that using a word embedding model learnt from a larger corpus helps. LSTM benefits most from GLOVE embedding, archiving the overall best performance with 48.58% accuracy.

11.3.2.4 Top most frequent answers

Our experiments reported in Table 11.3 investigate predictions using different number of answer classes. We experiment with a truncation of 1000, 2000, or 4000 most frequent classes. For all question encoders (and always using GLOVE word embedding), we find that

Method	Accuracy
BOW	53.27
CNN	54.23
GRU	54.23
LSTM	**54.29**

Table 11.5: Results on VQA validation set, vision and language: Analysis of different language encoders with GLOVE word embedding, VGG-19, and Summation to combine vision and language. Results in %, see Section 11.3.3.2 for discussion.

Method	Accuracy
AlexNet	53.69
GoogLeNet	54.52
VGG-19	54.29
ResNet-152	**55.52**

Table 11.6: Results on VQA validation set, vision and language: Different visual encoders (with LSTM, GLOVE, the summation technique, l2 normalized features). Results in %, see Section 11.3.3.3 for discussion.

a truncation at 2000 words is best, being apparently a good compromise between answer frequency and missing recall.

11.3.2.5 Summary Question-only

We achieve the best "Question-only" accuracy with GLOVE word embedding, LSTM sentence encoding, and using the top 2000 most frequent answers. This achieves an performance of 48.86% accuracy. In the remaining experiments, we use these settings for language and answer encoding.

11.3.3 Vision and Language

Although Question-only models can answer on a substantial number of questions as they arguably capture common sense knowledge, for further development we also need images.

11.3.3.1 Multimodal fusion

Table 11.4 investigates different techniques that combine visual and language representations. To speed up training, we combine the last unit of the question encoder with the visual encoder, as it is explicitly shown in Figure 11.1. In the experiments we use Concatenation, Summation, and Piece-wise multiplication on the BOW language encoder with GLOVE word embedding and features extracted from the VGG-19 net. In addition, we also investigate using L2 normalization of the visual features, which divides every feature vector by its L2 norm. The

	Question only			+ Vision
	Learnt -	GLOVE - word embedding		
Question encoding ↓	Top 1000 answers		Top 2000 answers	
BOW	47.41	47.91	48.13	54.45
CNN	48.26	48.53	48.67	55.34
LSTM	47.80	48.58	48.86	**55.52**

Table 11.7: Results on VQA validation set, vision and language: Summary of our results, results in %, see Section 11.3.4 for discussion. Columns denote, from the left to right, word embedding learnt together with the architecture, GLOVE embedding that replaces learnt word embedding, truncating the dataset to 2000 most frequent answer classes, and finally added visual representation to the model (*ResNet-152*).

	Test-dev				Test-standard			
Trained on	Yes/No	Number	Other	All	Yes/No	Number	Other	All
Training set	78.06	36.79	44.59	57.48	-	-	-	57.55
Training + Val set	78.39	36.45	46.28	**58.39**	78.24	36.27	46.32	**58.43**

Table 11.8: Results on VQA test set, our best vision and language model chosen based on the validation set: accuracy in %, from the challenge test server. Dash '-' denotes lack of data

experiments show that the normalization is crucial in obtaining good performance, especially for Concatenation and Summation. In the remaining experiments, we use Summation.

11.3.3.2 Questions encoders

Table 11.5 shows how well different questions encoders combine with the visual features. We can see that LSTM slightly outperforms two other encoders GRU and CNN, while BOW remains the worst, confirming our findings in our language-only experiments with GLOVE and 2000 answers (Table 11.3, second column).

11.3.3.3 Visual encoders

Next we fix the question encoder to LSTM and vary different visual encoders: Caffe variant of *AlexNet* [Krizhevsky et al. 2012], *GoogLeNet* [Szegedy et al. 2015], *VGG-19* [Simonyan and Zisserman 2015], and recently introduced 152 layered *ResNet* (we use the Facebook implementation of He et al. [2015]). Table 11.6 confirms our hypothesis that stronger visual models perform better.

11.3.3.4 Qualitative results

We show predicted answers using our best model on VQA test set in Tables 11.11, 11.12 ,11.13, 11.14. We show chosen examples with 'yes/no', 'counting', and 'what' questions,

	Test-dev				Test-standard
	Yes/No	Number	Other	All	All
DMN+ [Xiong et al. 2016]	80.5	36.8	48.3	60.3	60.4
FDA [Ilievski et al. 2016]	81.1	36.2	45.8	59.2	59.5
AMA [Wu et al. 2016b]	81.0	38.4	45.2	59.2	59.4
SAN(2, CNN) [Yang et al. 2015]	79.3	36.6	46.1	58.7	58.9
Refined Ask Your Neurons	78.4	36.4	46.3	58.4	58.4
SMem-VQA [Xu and Saenko 2015]	80.9	37.3	43.1	58.0	58.2
D-NMN [Andreas et al. 2016a]	80.5	37.4	43.1	57.9	58.0
DPPnet [Noh et al. 2015b]	80.7	37.2	41.7	57.2	57.4
iBOWIMG [Zhou et al. 2015]	76.5	35.0	42.6	55.7	55.9
LSTM Q+I [Antol et al. 2015]	78.9	35.2	36.4	53.7	54.1
Comp. Mem. [Jiang et al. 2015]	78.3	35.9	34.5	52.7	-

Table 11.9: Results on VQA test datasets, comparison with state-of-the-art: accuracy in %, from the challenge test server. Dash '-' denotes lack of data. The full table is shown in Malinowski et al. [2016].

where our model, according to our opinion, makes valid predictions. Moreover, Table 11.14 shows predicted compound answers.

11.3.4 Summary VQA results

Table 11.7 summarises our findings on the validation set. We can see that on one hand methods that use contextual language information such as CNN and LSTM are performing better, on the other hand adding strong vision becomes crucial. Furthermore, we use the best found models to run experiments on the VQA test sets: test-dev2015 and test-standard. To prevent overfitting, the latter restricts the number of submissions to 1 per day and 5 submissions in total. Here, we also study the effect of larger datasets where first we train only on the training set, and next we train for 20 epochs on a joint, training and validation, set. When we train on the join set, we consider question answer pairs with answers among 2000 the most frequent answer classes from the training and validation sets. Training on the joint set have gained us about 0.9%. This implies that on one hand having more data indeed helps, but arguably we also need better models that exploit the current training datasets more effectively. Our findings are summarized in Table 11.8.

11.4 State-of-the-art on DAQUAR and VQA

In this section, we first put our findings on VQA in a broader context, where we compare our refined version of *Ask Your Neurons* with other, publicly available, approaches. Next, guided by our findings on VQA, we re-run the experiments on DAQUAR.

	Accuracy on		WUPS@0.9 on		WUPS@0 on	
	all	single	all	single	all	single
Global						
Ask Your Neurons	19.43	21.67	25.28	27.99	62.00	65.11
Refined Ask Your Neurons	24.48	26.67	29.78	32.55	62.80	66.25
Refined Ask Your Neurons $*$	25.74	27.26	31.00	33.25	63.14	66.79
IMG-CNN [Ma et al. 2015]	21.47	24.49	27.15	30.47	59.44	66.08
Attention						
SAN (2, CNN) [Yang et al. 2015]	-	29.30	-	35.10	-	68.60
DMN+ [Xiong et al. 2016]	-	28.79	-	-	-	-
ABC-CNN [Chen et al. 2015]	-	25.37	-	31.35	-	65.89
Comp. Mem. [Jiang et al. 2015]	24.37	-	29.77	-	62.73	-

Table 11.10: Comparison with state-of-the-art on DAQUAR. Refined Ask Your Neurons architecture: LSTM + Vision with GLOVE and ResNet-152. Ask Your Neurons architecture: originally presented in Malinowski et al. [2015], results in %. In the comparison, we use original data (all), or a subset with only single word answers (single) that covers about 90% of the original data. Asterisk '$*$' after the method denotes using a box filter that smooths the otherwise noisy validation accuracies. Dash ''-' denotes lack of data.

VQA. Table 11.9 compares our *Refined Ask Your Neurons* model with other approaches. Some methods, likewise to our approach, use global image representation, other attention mechanism, yet other dynamically predict question dependent weights, external textual sources, or fuse compositional question's representation with neural networks. Table 11.9 shows a few trends: better visual representation helps, attention based models (e.g. *DMN+*, *FDA*, *SAN*) have a slight advantage over models with a global image representation (e.g. *Refined Ask Your Neurons*, *LSTM Q+I*), encoding longer dependencies in questions indeed helps (e.g. *Refined Ask Your Neurons* and *iBOWIMG*), using external textual resources is beneficial (*AMA*).

DAQUAR. Based on the VQA experiments, we have also applied the best model to DAQUAR significantly outperforming Malinowski et al. [2015] presented in Section 10.4. In the experiments, we first choose last 10% of training set as a validation set in order to determine number of training epochs K, and next we train the model for K epochs. We evaluate model on two variants of DAQUAR: all data points ('all' in Table 11.10), and a subset ('single word' in Table 11.10) containing only single word answers, which consists of about 90% of the original dataset. As Table 11.10 shows, our model, Vision + Language with GLOVE and Residual Net that sums visual and question representations, outperforms the model of Malinowski et al. [2015] by 5.05, 4.5, 0.8 of Accuracy, WUPS at 0.9, and WUPS at 0.0 respectively. This shows how important a strong visual model is, as well as the aforementioned details used in training. Likewise to our conclusions on VQA, we are

also observing an improvement with attention based models (comparison in *Attention* and *Global* sections in Table 11.10).

			
	Are the dogs tied?	Is it summer time?	Is this a real person?
Ask Your Neurons:	yes	no	no

Table 11.11: Examples of 'yes/no' questions and answers produced by our the best model on test VQA.

			
	How many kites are only yellow?	How many taps are on the sink?	How many windows are lit?
Ask Your Neurons:	1	2	12

Table 11.12: Examples of 'counting' questions and answers produced by our the best model on test VQA.

			
	What is the man holding to his ear?	What sport is this man enjoying?	What brand is the laptop?
Ask Your Neurons:	phone	snowboarding	apple

Table 11.13: Examples of 'what' questions and answers produced by our the best model on test VQA.

			
	Color of cow?	What is the man doing?	Where is the TV control?
Ask Your Neurons:	brown and white	playing wii	on table

Table 11.14: Examples of 'compound answers' questions and answers predicted by our the best model on test VQA.

Chapter 12

Conclusions and Future Perspectives

Contents

With the advent of large-volume datasets, and advances in Deep Learning, Visual Recognition has successfully changed its research direction from using hand-designed to learnt features. This move has resulted in many success stories in image classification and object detection tasks [Krizhevsky et al. 2012; Szegedy et al. 2015; He et al. 2015; Girshick et al. 2014; Ren et al. 2015b]. Observing such a progress, we ask in this thesis the following questions. Can we apply one of the highlights of Deep Learning, i.e. joint training, to a traditional recognition architecture (Spatial Pyramid Matching)? How to develop a system with the minimal number of design decisions? Can we build a neural approach to learn spatial relations that can easily scale up to handle a large number of spatial prepositions? Can we develop a neural network with an input-dependent structure? What is a good holistic task? How does the success of Deep Learning in Visual Recognition translate to other, arguably more holistic tasks? Can neural networks reason logically, and infer, often hidden, human intentions through communication with human beings? What is the relation between neural architectures and semantic parsers? In this chapter, we conclude our findings, and point a few possible research directions out.

12.1 Concluding Remarks

In this thesis, we build architectures that are jointly trained, reason about spatial relations, work in a multimodal scenario, answer questions about images, and to some extent unite vision with language through vector-based representations.

Joint training of a spatial layout and a classifier In Chapter 6, we jointly train a classifier together with a spatial pooling layer in a traditional recognition architecture, so called Spatial Pyramid Matching. Our results indicate that a-priori fixed spatial division of the Spatial Pyramid Matching architecture is indeed suboptimal, and hence should be learnt from data. However, the largest improvement, which reach up to 10 percent points compared with the baseline, is observed in the case of a limited number of visual words. This

effect may be due to that larger, more fine-grained visual vocabularies can preserve some spatial information, but this hypothesis is not further explored. Notably, in comparison with most other data-driven approaches to derive a spatial layout, our strategy rely on a minimal number of assumptions regarding shapes of the pooling regions. On the other hand, our smooth regularization term has been shown important.

Neural approach to learn spatial relations In Chapter 7, we link spatial templates with the aforementioned learnable spatial pooling regions. This allows us to build an architecture that spatially reason about the 'things' in the image. Unfortunately, we have never been able to learn such spatial templates from scratch, but instead we first estimate them from the training data and next use them as an initialization. Moreover, the learnt spatial templates do not differ that much from the initial estimations. On the other hand, with larger datasets, advances in Deep Learning, more powerful hardware, and optimized convolutions training such architecture could, arguably, be feasible. Another possible extension of this work would learn an image-dependent spatial reasoning. Nonetheless, our estimation has been shown very competitive to other methods on the dataset of structured queries containing two spatial prepositions [Lan et al. 2012] but working under weaker assumptions. That is, our method does not have an access to a ground-truth procedure of generating spatial content. We also show that our method can easily scale up to work with a larger number of spatial prepositions.

Neural approach to retrieval with an input-induced architecture In Chapter 7, we use Data-Driven Compositional Neural Architecture in some experiments. A detailed exposition of the architecture is presented in Appendix A. The structure of the presented method is induced from a textual query. More precisely, a parser tree defines the topology of the network, with leaves representing nouns, and internal nodes representing spatial relations. The weights are indexed by occurring words, and are shared across all the architectures. Alternatively, this can be seen as a large set of weights corresponding to all the words, while only a small subset of them is updated per training example. Such a parameterization on one hand allows for an efficient training, on the other hand weights that correspond to rare words are rarely (or even never) updated. In our work, we simplified the problem by considering only queries of a relatively simple form, e.g. (noun, preposition, noun). Therefore, it is interesting to see how the architecture extends to real-world textual queries. Interestingly, this architecture can also be seen as a mixture of a semantic parser used in Chapter 9 and a neural network used in Chapter 10.

Holistic tasks In many parts of this thesis – e.g. Chapters 8, 9, 10 – we argue for an alternative, holistic task for a scene understanding. Object detection or semantic segmentation, two examples of tasks requiring a scene understanding, have a few limitations. The annotation effort is big, especially costly are per-pixel annotations for the scene segmentation task, and both tasks unnecessarily impose some representation of an image (rectangular bounding boxes, or per-pixel labels). Our instantiation of the Visual Turing Test, which is the DAQUAR dataset, also differs from the image description task by limiting the output

space. This greatly simplifies the evaluation of architectures on this task, and yet imposes a great challenge to intelligent machines. As a further matter, as we argue in Malinowski and Fritz [2015], the task of answering questions about images is more focused than image description and arguably more robust to over-interpretations than the original Turing Test Turing [1950]. Interestingly, Visual Turing Test already encompasses many prior visual tasks. For instance, image classification can be seen as a Visual Turing Test with only one question "what is in the image?"; similarly, Visual Turing Test without questions can be interpreted as the image description task, where a caption (answer) describing an image has to be derived. Therefore, presumably, successful Visual Turing architectures can be deployed in numerous scenarios. Moreover, the task is inherently multimodal, and challenges machines not only in visual and language comprehension, but also in understanding human intentions and requiring a complex reasoning. Finally, such a question answering task is, arguably, a necessary step to achieve a full, human-quality competence in understanding of the world, and therefore should be a component of AI-complete systems.

Neural and symbolic approaches In this thesis, we propose two approaches to answer questions about real-world images that are introduced in Chapters 9 and 10. Both mark two extremes on a spectrum of different methods ordered with respect to explicitness in image and language representations (Figure 5.3). A logic-based approach relies on a semantic parser and a database of visual facts. Therefore, the overall performance of the system strongly depends on the choices made by a designer who decides what information is being extracted and how the spatial relations are defined. At it turns out, it is not only a highly non-trivial task, but also errors made at this fundamental level of defining a right representation cannot be easily corrected [Malinowski and Fritz 2014a,b; Chowdhury et al. 2016a]. Moreover, such an approach relies too much on the visual information, ignoring a knowledge that comes from the linguistic channel. To alleviate such issues, we have proposed our second approach, which we call a neural-based, that relies on a combination of a Convolutional Neural Network and a Recurrent Neural Network. This architecture requires fewer design decisions, and can learn a representation of the input data for the task jointly and end-to-end. It can also effectively exploit information existing in the language channel. On the other hand, we observe little or even no evidence it can handle questions requiring a logical notion, e.g. negations. It also relies too much on the language channel making only a moderate improvement when a visual channel is added (Table 10.1).

Representation In the majority of this thesis, we adopt the Deep Learning point of view on the representation, and therefore we rather rely on learnt representations. For instance, in Chapter 6, Chapter 7, and Appendix A, we argue for a representation that is directly derived from the training data, ideally in the gradient-based manner. In Chapter 10, Chapter 11, and Appendix C, we build a Deep Learning approach to answer questions about real-world images. We made a major exception in Chapter 9, where we trained a logic-based approach to answer question about real-world images that rely on hand-designed representation of the world. More precisely, we have designed a set of spatial rules, and decided what kind of visual information shall be extracted from images. In our research, a learnt representation

has always achieved better performance. Such results are also consistent with a broader research done in Computer Vision where the leading approaches to many Computer Vision tasks use learnt representations.

Scalability In many practical scenarios we seek scalable architectures that are able to learn more as data grow, and quickly perform an inference at test time (Section 2.1). In literature, we can find some evidence that neural networks are good in exploiting large-volume datasets [Krizhevsky et al. 2012]. They have also dominated the VQA challenge [Antol et al. 2015; Malinowski et al. 2016] that consists of one of the largest Visual Turing datasets. Moreover, such neural-based architectures can process a large amount of data in a short time on a high-throughput hardware. For instance, Caffe, a popular Deep Learning toolkit, spends about 1ms per image to derive an image representation[1]. This contrasts with a logic-based approach, shown in Chapter 9, that requires first extracting information from images to build a database of visual facts, and next deriving a suitable representation of a question by manipulating potentially exponentially many derivations. Therefore, apparently it is more difficult to use a semantic parser on large-volume datasets, although there is a notable progress in this direction [Berant et al. 2013; Choi et al. 2015].

Joint, end-to-end training In this thesis, we mainly advocate for jointly trained architectures via back-propagation. This view is covered in the many chapters of the thesis (Chapters 6, 7, 10, 11). The same view also becomes dominant in the whole Computer Vision community. Through the thesis, we also observe an increase of the performance whenever a jointly trained scheme is used. For instance, our variant of the Spatial Pyramid Matching framework (Section 3.2) is able to discriminatively learn a spatial layout based only on data, and significantly improves over the original architecture with a hand-designed spatial layout (Chapter 6). Similarly, our neural-based approach to Visual Turing Test (Chapter 10) significantly outperforms our prior work that uses a logic-based one (Chapter 9).

12.2 Future Perspectives

This section discusses various items of future work based on different research directions taken in the thesis. We pay a special attention to our most recent work on the Visual Turing Test, which has opened the door to many new ideas. Therefore the majority of future directions are based on our experience while working on the aforementioned problem.

Variants of Visual Turing Test Visual Turing Test encompasses various tasks such as detection or classification. Due to the generality of the Visual Turing Test, various datasets that stress different aspects of the challenge have been proposed. For instance, in DAQUAR spatial relations play an important role [Malinowski and Fritz 2014a], while Zhu et al. [2016] also allows questions about locations of different objects or their parts (so called 'pointing' questions), and Kembhavi et al. [2016] have built a dataset for question answering about diagrams. In the nearest future, we may expect more variants of the Visual Turing Test,

[1] On a single NVIDIA K40 GPU: http://caffe.berkeleyvision.org

some could be even more generic than the contemporary datasets, and some could encourage studies of specific aspects in isolation. For instance, synthetic datasets with compositional questions could test if machines can ground difficult questions into images.

Common sense knowledge Common sense knowledge plays an important role in the human decision making process. Arguably, holistic machines also need to be equipped with the common sense [Malinowski and Fritz 2014b]. However, extracting such a knowledge only from textual sources is a challenging task, and therefore alternative approaches to acquire it through multimodal, visual and language, datasets is not only possible [Malinowski et al. 2015], but even become an active research area [Chowdhury 2016; Chowdhury et al. 2016b; Tandon 2016; Vedantam et al. 2015; Sadeghi et al. 2015]. Nonetheless, we believe in a tighter interplay between the Visual Turing Test and common sense knowledge.

External sources of knowledge Not all questions in the Visual Turing Test can be grounded in an image. Some of them can only be answered through external sources of knowledge such as Wikipedia. For instance, questions such as 'What city is visible in the image?' requires a more holistic knowledge about the world. Work of Wu et al. [2016b], and Wang et al. [2015] is a notable step towards this direction.

Multimodal fusion Despite of the tremendous progress in visual recognition, where the most recent approaches achieve human-quality performance on the ImageNet dataset [Russakovsky et al. 2014; He et al. 2015], the overall performance of the current approaches is still far from a human-quality understanding. We believe that this is partially due to the lack of proper multimodal fusion techniques that combine vision with language. The contemporary fusion techniques either concatenate, piecewise multiply, or sum vector representations of the question and the image. However, we arguably need more sophisticated tools to combine both modalities together. A step into this direction is the recent work of Fukui et al. [2016] that uses the outer product as a fusion technique. This technique allows for multiplicative interactions, where each element can interact with every other element. Since this approach has won the VQA challenge, and a novel multimodal fusion technique is an important factor that contributed to its success, we believe that a further development of the multimodal fusion techniques is crucial for the Visual Turing Test.

Spatial reasoning A substantial fraction of questions in DAQUAR – our dataset that represents the Visual Turing Test – consists of spatial prepositions. Such questions are also nontrivial to answer even to humans due to different sources of ambiguities; e.g. depending on personal or cultural preferences humans use different frames of references [Malinowski and Fritz 2014a]. Despite of the initial progress to handle the spatial relations [Malinowski and Fritz 2014c; Chowdhury et al. 2016a], current results are still somehow dissatisfying. That is, the performance of our methods on spatial questions is significantly below the overall performance even though such questions are quite representative in the DAQUAR dataset. Therefore, we hope that better approaches to spatial reasoning will be developed.

Image representation In this thesis, our neural-based approaches to Visual Turing Test use global, full-frame CNN representation of images. Such a representation may destroy too much information. This points towards a direction of fine-grained alternatives, such as detections or object proposals [Ilievski et al. 2016; Mokarian Forooshani et al. 2016; Tommasi et al. 2016]. At the same time, recently introduced models with attention become quite successful [Fukui et al. 2016; Lu et al. 2016]. However, it is unclear if the gap between attention-based and global-based representations is due to the additional constraints imposed on the attention mask, or unsuccessful, joint, end-to-end training of the methods that use the global image representation. Moreover, it is also surprising that approaches to Visual Turing Test that use detections do not perform significantly better to the aforementioned methods. Finally, DAQUAR contains images with a depth channel that is currently not leveraged by neural-based methods. Some spatial relations, such as 'behind', arguably, require a 3d scene representation. The question of the importance of such an extra information for the Visual Turing Test remains open. To sum up, it is still unclear which image representation should be used to develop future approaches to the Visual Turing Test.

Recurrent Neural Networks Although Recurrent Neural Networks should be better than Bag-Of-Words approaches in modeling a complex language structure, there is a surprisingly small gap in the performance between both approaches [Malinowski et al. 2016]. However, some questions clearly require an order. At the same time, such questions are longer, semantically more difficult, and require better visual understanding of the world. To handle such questions we may need to improve over the current Recurrent Neural Networks, find better ways of fusing two modalities, develop better image representations, or build datasets that stress compositionality of the language more clearly.

Evaluation metrics Although we have WUPS and Consensus measures at our disposal, both metrics are far from being perfect. Consensus has higher annotation cost for ambiguous tasks, and is unclear how to formally define a good consensus measure. WUPS is an ontology-dependent evaluation metric, and hence it rises the question if building one complete ontology that covers all cases is feasible. Moreover, the currently used evaluation metrics do not consider the tail of the distribution of answers, and hence encouraging the methods to focus only on the most frequent answers. Overall, developing good evaluation metrics in a nontrivial task that requires some attention.

Learning from a few examples In the DAQUAR dataset, many questions are quite unique. This raises a challenge on the generalization of the methods. On the other hand, questions are also, to some extent, compositional. Can we build models that leverage such compositionality in order to build the meaning of the whole question from its parts? Shall we explicitly model the compositionality of the meaning [Liang et al. 2013; Malinowski and Fritz 2014a], or leave it as a learning task [Malinowski et al. 2015]? Similarly, the contemporary models tend to focus on the most frequent answers, ignoring the tail of the answer distribution. This is arguably not a desired behavior from the intelligent machines.

CNNs with smooth pooling regions Our investigation of the smooth learnable pooling regions have shown good results on the more classical recognition architecture – Spatial Pyramid Matching, while allowing for introspections by visualization. Despite of differences between Convolutional Neural Networks and Spatial Pyramid Matching, both architectures use a pooling operation over its receptive fields. As we did with Spatial Pyramid Matching, the spatial layout could also be learnt or estimated from data for Convolutional Neural Networks. Technically, the learnable pooling regions are achieved by a suitable generalization of the pooling operator together with adding the smoothness-inducing regularization terms. Finally, the learnt pooling regions also reminds the attention masks that become commonly used in neural image description or question answering architectures [Xu et al. 2015; Xu and Saenko 2015]. We also observe an interest of the community in that direction of research [Yang et al. 2016; Lee et al. 2016; Fukui et al. 2016], and we believe the pooling operator will undergo significant changes in future.

Personalized question answering architectures Due to many ambiguities that exist in the real-world, also reflected in some Visual Turing Test datasets such as DAQUAR [Malinowski and Fritz 2014a], it is difficult to train models to handle all the questions properly. This suggests personalized approaches to Visual Turing Test, where the methods are conditioned on the specific user. Chowdhury et al. [2016a] further study this problem based on which they propose a few possible directions.

Connections with physics As we are pushing the boundaries of Computer Vision, understanding of physics (at least at the intuitive level [McCloskey 1984]) becomes a key component of the holistic comprehension. It could be the case that some questions about the content of images cannot be resolved without a basic understanding of physics. Currently, we witness initial steps towards this direction by considering scenarios fully specified by physics [Battaglia et al. 2013; Wu et al. 2015; Mottaghi et al. 2016; Fragkiadaki et al. 2015], or scenarios where physics play an important but latent role and can, arguably, be internalized by a neural architecture [Bhattacharyya et al. 2016a,b].

Synergy of neural-based approaches with symbolic reasoning Although we have observed a remarkable performance improvement of our neural-based approach over our prior work that uses a semantic parser (Chapters 9 and 10 contain more details about both approaches to the Visual Turing Test), some types of logical reasoning seems to be particularly difficult for the former while trivial for the latter. One example is the negation that is 'built' into the semantic parser and therefore handling negations with such an approach at the semantic level is trivial. However, based on the current evidence we cannot conclude if the neural-based approach learns to negate the question or its parts. Similarly, we have not observed that it successfully resolves quantifications. At this point, it is hard to conclude if there is something fundamental missing or it is just due to lack of enough training data. However, we believe that this line of research deserves a more thorough investigation, which may result in hybrid models. Our DDCNA (Appendix A) for the text-to-image retrieval task, and especially the work of Andreas et al. [2016b] for the visual question

answering task can be seen as a step forward in that direction.

Appendix A

DDCNA: Data-Driven Compositional Neural Architecture for Image Retrieval based on Compositional Queries

Contents

This appendix supplements Chapter 7. It presents a preliminary work, which precedes Visual Turing Test and was conducted in 2013-2014, that study a compositional nature of spatial relations with a neural-based architecture with an instance-induced topology. The structure of this architecture, which we call DDCNA, is inferred from a textual query, and therefore its topology is changing per instance. In this sense, each training and test instance poses a different task. Finally, in this architecture, we stress concept learning, and spatial reasoning.

A.1 Introduction

Recent advances in object classification and detection [Felzenszwalb et al. 2010; Li et al. 2010; Krizhevsky et al. 2012] have significantly contributed to the progress towards the challenging

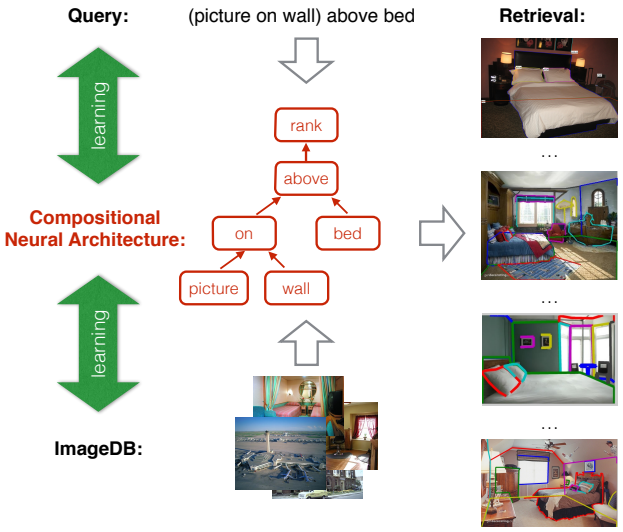


Figure A.1: We address the image retrieval task by introducing a novel Data-Driven Compositional Neural Architecture (DDCNA) whose topology is induced from the query. The parameters – including concepts and spatial relations – are shared across queries and jointly learnt with the retrieval task.

task of scene understanding [Lin et al. 2013; Gupta et al. 2013]. Equally, recent methods for image retrieval [Lan et al. 2012] build on this success by forming queries of single binary spatial relations and incorporating strong object detectors in their framework. Unfortunately, such retrieval systems are still working with rather simple queries or with hand-crafted features to learn spatial relations, where it remains unclear how to extend the language of spatial relations to new prepositions. In contrast, we envision a data-driven architecture for image retrieval task with more complex textual queries that learns a representation for spatial relations directly from data.

In order to facilitate adaptation, ease of deployment and optimal leverage of the visual classifiers, we seek a system that is jointly trained from example queries only. This implies that there is no explicit supervision for the association of the visual classifiers to the language concepts as well as no hand-crafted or annotated information for spatial relations. Moreover, in our work we investigate jointly learnt representation of the concepts and spatial relations. For our study, we assume that the query is given in a tree structured form as shown in Figure A.1.

We address this challenging learning problem by a novel type of neural networks that we call Data-Driven Compositional Neural Architecture (DDCNA). For our application to image retrieval, these networks are trained from pairs of queries and images. As outlined in Figure A.1, the query induces the architecture of the network. In contrast to recent recursive neural networks [Socher et al. 2011], they are not composed of a single reoccurring network fragment, but rather of a set of fragments that are shared across all examples. Each fragment is associated to exactly one part of the query. We show how to perform holistic training of this network and give insights to the learnt parameters that capture refined notion of spatial relations.

Problem statement In our work we have a set of images $\mathcal{I}$ and a set of queries $\mathcal{Q}$. All queries are built of the nouns representing objects and spatial prepositions such as 'above', 'left of', 'under', etc. Next, every query q is represented by a parse tree $\mathcal{T}(q)$ [Manning and Schütze 1999]. Our task is to find a mapping $\mathcal{F}$ from $\mathcal{T}(q)$ into a set of relevant images to the query $\mathcal{I}(q) \subset \mathcal{I}$ for every query q.

Our contributions We propose a novel discriminatively trained neural architecture for image retrieval from textual queries that we call Data-Driven Compositional Neural Architecture (DDCNA). Our method can deal with more complex queries than previous approaches [Siddiquie et al. 2011; Lan et al. 2012]. Previous work employes hand-crafted spatial relations [Moratz and Tenbrink 2006; Kelleher et al. 2006] or more recently learns relations based on a hand-crafted set of features [Golland et al. 2010; Lan et al. 2012]. In contrast, our approach is founded on a pooling interpretation of spatial relations [Logan and Sadler 1996] with the goal of liberating the architecture from manually designed features. Therefore a rich set of relations can be learnt as convolutional filters. We extend an existing image retrieval benchmark based on the SUN09 dataset [Lan et al. 2012] with queries containing more spatial prepositions such as 'left of', 'right of', 'in front of', 'behind', 'inside of', 'on', 'under', 'across from' and 'in'. Moreover, we also add complex, compositional queries (e.g. 'picture on the wall above a bed'). We experimentally show that our architecture performs on par with previously proposed method [Lan et al. 2012], but without relying on hand-crafted representation of the spatial relations. We further highlight the learning-based approach of our method by showing results on the new extended dataset that features new spatial relations and complex, composite queries that have previously not been addressed [Lan et al. 2012].

A.2 Related work

Modeling spatial relations in images Previous work has investigated hand designed features in order to ground spatial language e.g. for video search [Tellex et al. 2010]. Others have addressed the problem of image retrieval with structured object queries [Lan et al. 2012] where, in contrast to previous work in image retrieval, the authors consider structured queries - a textual input with binary spatial relations between objects. Our work considers not only binary relations but also compositions of those. Furthermore, we aim for an open vocabulary of the preposition available during training and therefore instead of using a hand-crafted representation of a set of only few relations ('above', 'below', and 'overlap' like in Lan et al. [2012]), we propose a flexible and learnable representation of such spatial relations based on a pooling interpretation of spatial relations [Logan and Sadler 1996].

Program induction for question answering Our work is partly inspired by Liang et al. [2011] who try to find a good semantic representation where utterances are mapped into their meanings. However, Liang et al. [2011] consider the problem of finding such representation only from the linguistic perspective, through the mapping into logical forms and retrieving answers from a knowledge base. In contrast, our work induces a neural architectures that

solves an inference task on natural images from textual queries.

Deep architectures Although our architecture has some resemblances to deep neural networks [Krizhevsky et al. 2012; Vincent et al. 2008], it differs by inducing the network architecture from the query. Therefore, the topology of our architecture is not fixed a priori to the dataset but is different for each training/test example. In this aspect it shares similarities to Socher et al. [2011], but differs in: latent and learnt representation of spatial relations, a hierarchical network structure that is composed of multiple shared fragments instead of one, joint training of the whole tree-based architecture as opposed to learning by greedily merging subtrees.

Image summarization An interesting inverse problem to our task is summarization of the images [Farhadi et al. 2010; Kulkarni et al. 2011], where text is retrieved based on the visual query - an image. It is however unclear how to use such architectures for our task.

Spatial reference resolution In contrast to previous attempts to spatial reference resolution and concept acquisition [Socher et al. 2000; Matuszek et al. 2012], we go beyond a set of object instances or toy objects to object categories in real-world image data.

A.3 Method

In this section we present our approach, Data-Driven Compositional Neural Architecture (DDCNA), for learning a representation from image-query pairs for image retrieval. In contrast to other methods common in deep learning community, the topology of our architecture is determined by the query. As opposed to recursive neural networks [Socher et al. 2011], it shares not a whole layer but the parameters associated to the same word across different instances of the network. The leaves of the resulting architecture correspond to visual concepts and internal nodes are relations between them. While the visual concepts are modeled as linear combinations of object detectors, we represent spatial relations as convolutional filters, that relate to a pooling interpretation of spatial relations [Logan and Sadler 1996]. The root is a classifier that output is used for the retrieval task. We learn the shared parameters of the network directly on image-query pairs in order to optimize performance on the image retrieval task. There is no explicit supervision for the individual network fragments that associate concepts as well as spatial relations - both are treated as latent.

A.3.1 Data-Driven Compositional Neural Architecture

Given the query-image pairs as input (q, i) we define an architecture $\mathcal{F}(q, i) := \text{DDCNA}(\mathcal{T}(q), \mathcal{I}(i)) \in [0, 1]$ that measures the relevance of the image i to the query q. We aim at learning such architecture solely from $\left\{(q^k, i^k), y^k\right\}_{k=1}^{N}$ where $y^k \in \{0, 1\}$ is a k-th retrieval label indicating relevance of the image i^k to the query q^k. Mapping $\mathcal{F}$ consists of a tree structured representation $\mathcal{T}(q)$ of the query, an image representation $\mathcal{I}$, and DDCNA - our new deep architecture.

As shown in Figure A.1, the topology of DDCNA is defined over a tree inferred by $\mathcal{T}$ from the query q consisting of three node types: leaves representing concepts, internal nodes that combine the concepts via a spatial relation, and a root node predicting a score which is the relevance of the image w.r.t. a query q. During training, we learn the representation of the concepts from $\mathcal{I}(i)$ together with the representation of the spatial relations and a classifier's weights to output the final score. During inference, we start from the leaves of the tree and next propagate the representation up to the root of the tree via the spatial relations that combine representations of theirs children.

Formally, we recursively define our architecture as follows. Let $X^{\text{word}}_{\text{leaf}}$, $X^{\text{word}}_{\text{internal}}$ be the leaf and internal nodes representing words, and X_{root} be a root node - output of DDCNA. Let $\mathcal{I}(i) := \{\mathcal{O}_k\}_k$ be a representation of the content of the image. We define a concept $\mathcal{O}^{\text{word}}$ that is associated with a word as a weighted combination: $\mathcal{O}^{\text{word}} := \sum_k w^{\text{word}}_k \mathcal{O}_k$. Similarly, we have $\mathcal{R}^{\text{word}} := \sum_k w^{\text{word}}_k \mathcal{R}_k$ to represent spatial relations for the spatial preposition (a word). Both sets $\{\mathcal{O}_k\}$ and $\{\mathcal{R}_k\}$ can be seen as basis, and we train the architecture via backprop by learning the parameters w^{word}_k. The concepts are associated with the leaves, while the spatial relations with the internal nodes. Following the tree structure, we have

$$\begin{aligned} X^{\text{word}}_{\text{leaf}} &:= \mathcal{O}^{\text{word}} \\ X^{\text{word}}_{\text{internal}} &:= X_{\text{subtree}_1} \circ \left(\mathcal{R}^{\text{word}} * X_{\text{subtree}_2}\right) \\ X_{\text{root}} &:= \mathcal{J}(\text{pool}(X^{\text{query}}_{\text{internal}})) \end{aligned} \tag{A.1}$$

where $\mathcal{J}$ is a classifier and "pool" is a pooling operator, and X_{subtree} is either a leaf or an internal node. We also use two operators to combine the subtrees: piece-wise multiplication $\circ$ and convolution $*$. During training, we learn the parameters $w : \mathcal{L} \to \mathbb{R}^K$ for a vocabulary $\mathcal{L} := \{\text{word}_1, \text{word}_2, ...\}$. Note that our architecture is conditioned on the query representation and therefore instance-dependent.

In our study, we use a parse tree of the query q as its representation $\mathcal{T}(q)$ [Manning and Schütze 1999; Klein and Manning 2003] and an over-complete set of the object detections [Felzenszwalb et al. 2010; Li et al. 2010] to represent the content of the image $\mathcal{I}(i)$. We use Dirac responses (a single element of an A-by-B matrix is active) as spatial basis $\{\mathcal{R}_k\}_{k=1}^{A \cdot B}$. Moreover, we employ logistic regression $\mathcal{J}(X) := \mathbb{E}\left[\{y = 1\} \log(h_\theta(X)) + \{y = 0\} \log(1 - h_\theta(X))\right]$ with the hypothesis space $h_\theta(x) := \frac{1}{1+\exp(-\theta^T x)}$ for classification, and max-pooling as the pooling operator.

A.3.2 Inference

Inference is both an inherent component of the learning process and a procedure yielding predictions from the input, and is defined over a parsing tree of the query instance. In the inference process, during the tree traversal, two operators are applied to combine the internal representations:

- Piecewise multiplication $\circ$ to intersect two internal representations.
- Convolution $*$ to convolve the spatial representation with the internal representation.

where an internal representation is either the output of a leaf (then it is the same as concept representation) or an internal node. Intuitively, at every pixel p, convolution of the concept

$\mathcal{O}^{\text{word}_a}$ with spatial filter $\mathcal{R}^{\text{word}_c}$ computes the confidence that $\mathcal{O}^{\text{word}_a}$ is related with p via $\mathcal{R}^{\text{word}_c}$. Intersecting p with another concept $\mathcal{O}^{\text{word}_b}$ yields the confidence that both concepts are related via the spatial relation at pixel p. This idea of convolving the object with a spatial filter is a computational realization of the spatial templates proposed in experimental psychology [Logan and Sadler 1996].

To illustrate the inference, let us follow Figure A.1 together with the query 'Picture on wall, above bed'. First, we collect the representation of the image by running object detectors $\{\mathcal{O}_k\}$. Those are subsequently linearly combined to build a concept representation for all noun words in the query: picture, wall and bed. That is, $X^{\text{picture}}_{\text{leaf}} := \sum_k w^{\text{picture}}_k \mathcal{O}_k$, $X^{\text{wall}}_{\text{leaf}} := \sum_j w^{\text{wall}}_k \mathcal{O}_k$, and $X^{\text{bed}}_{\text{leaf}} := \sum_j w^{\text{bed}}_k \mathcal{O}_k$. Note that, the learnt parameters w^{picture}_k, w^{wall}_k, w^{bed}_k are shared across all queries containing picture, wall and bed respectively. Such a representation of the concept 'wall' is next convolved with the spatial filter 'on' and subsequently intersected with the concept 'picture': $X^{\text{picture on wall}}_{\text{internal}} := X^{\text{picture}}_{\text{leaf}} \circ \left(\mathcal{R}^{\text{on}} * X^{\text{wall}}_{\text{leaf}}\right)$. Similarly, a higher level internal representation is computed: $X^{\text{query}}_{\text{internal}} := X^{\text{picture on wall}}_{\text{internal}} \circ \left(\mathcal{R}^{\text{above}} * X^{\text{bed}}_{\text{leaf}}\right)$. The final representation is produced by pooling $x := \text{pool}(X^{\text{query}}_{\text{internal}})$. At the final step such the representation x is given to the classifier $\mathcal{J}$.

A.3.3 Learning

Training via Backpropagation As we aim for the joint training of the whole architecture, we use backpropagation [LeCun et al. 1998a] through the tree structure to learn the parameters of our model. The backprop rules mimic the rules of the forward pass (Equation A.1), but are executed from the top layers towards the leaves:

$$
\begin{aligned}
\nabla_{w^{\text{word}}} X_{\text{root}} &= \left(\nabla_Z J(Z)\right) \theta^T \left(\nabla_{w^{\text{word}}} X^{\text{query}}_{\text{internal}}\right) \\
\nabla_{w^{\text{word}}} \left[X_{\text{subtree}_1} \circ \left(R^{\text{word}} * X_{\text{subtree}_2}\right)\right] &= \nabla_{w^{\text{word}}} X_{\text{subtree}_1} \circ \left(R^{\text{word}} * X_{\text{subtree}_2}\right) \\
&\quad + X_{\text{subtree}_1} \circ \nabla_{w^{\text{word}}} \left(R^{\text{word}} * X_{\text{subtree}_2}\right) \\
\nabla_{w^{\text{word}_a}} \left(R^{\text{word}_b} * X_{\text{subtree}}\right) &= 0 \text{ if } \; a \neq b \\
\nabla_{w^{\text{word}_a}} \left(R^{\text{word}_a} * X_{\text{subtree}}\right) &= \left(\nabla_{w^{\text{word}_a}} R^{\text{word}_a}\right) * X_{\text{subtree}} \\
&\quad + R^{\text{word}_a} * \left(\nabla_{w^{\text{word}_a}} X_{\text{subtree}}\right) \\
\nabla_{w^{\text{word}_a}} X^{\text{word}_b}_{\text{leaf}} &= 0 \text{ if } \; a \neq b \\
\nabla_{w^{\text{word}_a}} X^{\text{word}_a}_{\text{leaf}} &= [\mathcal{O}_j]_j \text{ if } \; X^{\text{word}_a}_{\text{leaf}} = \sum_j w^{\text{word}_a}_j \mathcal{O}_j,
\end{aligned}
\tag{A.2}
$$

where $Z := \theta^T X$, notation $[Z_l]_l$ denotes the vector expansion of Z with respect to index l, that is $[Z_l]_l := [Z_1, Z_2, ..., Z_K]$, and $X_{\text{left}} \circ [Z_l]_l \circ X_{\text{right}}$ is re-defined as $[X_{\text{left}} \circ Z_l \circ X_{\text{right}}]_l$.

Although at first glance the gradient $\nabla_{w^{\text{word}}}$ needs to 'travel' over all children paths of a given node, in practice the same word rarely occurs more than once in a query and therefore the gradient for the associated parameter. Therefore, most parameters are not updated in a single stochastic gradient step.

For training, we have applied stochastic gradient descent method [Bottou 2012] with a constant learning rate:

$$W := W - \alpha \nabla_W \mathcal{F}_W(X)$$

where W are all parameters to learn, and $\mathcal{F}_W(X)$ is the objective function dependent on W and training data X. We use a GPU-based implementation based on Theano [Bergstra et al. 2010] in order to speed up the inference and training process.

Initialization Previous work has pointed out the importance of a proper initialization [Erhan et al. 2010; Çaglar Gülçehre and Bengio 2013]. Since our architecture has a very rich parameterization we seek a good initialization for the spatial filters. For every spatial preposition we consider all images that are valid under the preposition[1], for instance 'above'. Next, we consider all combinations of the objects occurring in the image that are consistent with its annotations, for instance 'picture above bed'. We center the spatial filter at every position inside the 'bed' detector and copy the content of the 'picture' detector.

A.4 Experiments

We conduct experiments on two challenging datasets. The first dataset is introduced by Lan et al. [2012] and contains structured queries that augments the SUN09 dataset. The second dataset is our proposed extension of the first dataset with more complex, compositional queries as well as a wider range of spatial relations. In this section, we describe our dataset, explain the experimental protocol, and show and discuss results.

A.4.1 Dataset

Images All our experiments are based on the real-world image material from the SUN09 dataset [Choi et al. 2010]. This dataset consists of $12,000$ annotated images with more than 200 object categories, and $152,000$ annotated object instances. We use 4367 images for training and 4317 images for testing - the same split as in Choi et al. [2010]. In our experiments, we use simple *structured* as well as more complex *compositional* queries that augment the SUN09 dataset.

Structured Queries The structured queries are introduced in Lan et al. [2012], but are not formally defined. Here, we formalize the notion of structured queries. We say that a query q is structured if it has the form: $q := q_1 \wedge q_2 \wedge ... \wedge q_n$, where q_i denotes either a noun or a triplet (noun, preposition, noun).

Compositional Queries In order to consider a richer set of natural language queries, in addition to the structured queries we also use compositional queries. We identify a set of compositional queries $\mathcal{Q}$ with queries that can be represented by a syntactic tree. Formally, $\mathcal{Q} := \{\mathcal{T}(q) \mid q \in \mathcal{L}\}$, where $\mathcal{L}$ represent a set of all queries in natural language, and $\mathcal{T}(q)$ is a syntactic tree of q. Compositional queries subsumes structured queries as a special case. In this paper, we work with a subset of such compositional queries of the form [(noun, preposition, noun), preposition, noun].

[1] Such information is available during training time anyway as it is needed for training the classifier.

New dataset of complex, compositional queries Although the SUN09 dataset has been originally developed for the scene recognition and object detection, we augment the dataset with textual queries. We use the structure (a) queries from Lan et al. [2012] of form (noun, spatial preposition, noun) with spatial prepositions such as 'above', 'below' and 'overlap'. However, in order to show that our architecture is capable of learning new relations we have extended the set of queries to 'left of', 'right of', 'in front of', 'behind', 'inside of', 'on', 'under', 'across from' and 'in'. Moreover, we also extend the set of queries to have a compositional form [(noun, spatial preposition, noun), spatial preposition, noun] such as 'car on road, behind truck', or 'picture on wall, above bed' (Figure A.1). We collect annotations of such type by first asking participants to provide descriptions of randomly selected images from the SUN09 dataset using these prepositions and compositional queries. In the second pass we curate this dataset and arrive at 53 structured queries and 15 compositional queries. Finally, the annotators annotate the relevance of each training and test image according to all queries. As the latter requires a lot of human effort we have automatized the process by showing only those images that contain all objects described in a query. In this process we have collected in total 68 different queries with a total of 590512 annotations. Our dataset is challenging due to the use of spatial relations by the annotators, variations of object appearance in real-world images in the SUN09 dataset, and the compositional form of the queries. Latter makes it intractable to collect a sufficient number of data points that is large enough to cover all possible queries due to its combinatorial complexity, and therefore requires algorithms that can efficiently re-use learnt substructure.

A.4.2 Evaluation

We have investigated three experimental settings. First, we compare our method against previous work on structured queries [Lan et al. 2012], where we show results of our method alone and together with pairwise terms. Second, we use the proposed dataset of extended structured queries and show generalization to new spatial relations. Finally, we have measured the performance of our method on the new complex, compositional query dataset, which go beyond the scope of the previous work. In all experiments we use Mean Average Precision (mAP) across all queries to measure the performance of different methods.

Comparison to previous work on structured queries In order to establish a comparison to previous work on structured queries, we run experiments on the structured queries from Lan et al. [2012] and compare to their approach in Table A.3. This dataset consists of 862 (463 for training and 399 for testing) queries of the form (noun, preposition, noun) with 111 nouns and only two different spatial relations ('above' and 'below'). In this dataset, the spatial relations are automatically extracted by a predefined formula on the (x, y) coordinates of objects and serve as exact definitions of the spatial relations. This procedure is also used by the system of Lan et al. [2012]. In contrast, we assume the procedure is unknown to our system as we are aiming at learning good spatial representations only from data. The structured model of Lan et al. [2012] that implements a structured SVM approach and models both the spatial relationship between objects in the query and co-occurrence between non-query and query objects achieves a performance of 11.16% mAP. Moreover, we also

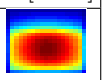
Lan et al. [2012]	
	
above	below

Table A.1: Visualization of estimated spatial filters. A set of relations from Lan et al. [2012].

Extended set of relations										
										
above	across from	behind	below	in	in front	inside	left	right	on	under

Table A.2: Visualization of estimated spatial filters. Extended set of relations.

report the results of two more baselines: simple object detector where the sum of maximum response scores from each object detector is used as a score and the MARR model [Lan et al. 2012]. The latter uses object detectors as the features for the classifier and models co-occurrence between the detectors.

In order to get more insights about the proposed model, we use five different variants of the architecture. First, 'DDCNA init' denotes our architecture with the estimated spatial filters and initialized concepts with only the logistic regression layer trained. We achieve 8.48% mAP for this setting which is an improvement over the 'Part based detector' by 0.72 percent points of mAP. Next, 'ßDDCNA with learnt spatial filters' uses 'DDCNA init' but also backprop to fine-tune the spatial filters which results in a further improvement of 0.18 percent points of mAP. The 'DDCNA with learnt concepts' trains both the classifier together with the concepts, which increases performance by 1.45 percent points. Training spatial relations, concepts and the classifier jointly yields in further improvements reaching 10.04% mAP ('DDCNA full'). Finally, the 'DDCNA full + pairwise terms' uses 'DDCNA full' as a feature vector for the classifier, and models co-occurrence between the object detector using equation:

$$\sum_{i \in V_q} \alpha_i^T f(I(l_i)) + \sum_{i \in V_q} \sum_{j \in \mathcal{X} \setminus V_q} \gamma_{ij}^T f(I(l_j)) + \beta^T \mathcal{F}(I_q) \tag{A.3}$$

where α, γ and β are weights learnt by the classifier, V_q is a set of all objects (nouns) in the query, $\mathcal{X}$ is a set of all objects available during training, $\mathcal{F}(I_q)$ is 'DDCNA init' for the query q, and $f(I(l_i))$ is a detector for the object i. We use the last variant in order to be as close as possible to the setting of Lan et al. [2012]. We reach a performance of 11.12% mAP (DDCNA full + pairwise terms) which is on par to the performance of the structured model [Lan et al. 2012] (Structure model in Table A.3), while not using the pre-defined notion of spatial relations and rather learning it from data. The two entries in Table A.1 show the filters that we have learnt to capture a notion of the spatial relations.

Method	mAP
Part based detector [Felzenszwalb et al. 2010]	7.76%
MARR [Siddiquie et al. 2011]	10.01%
Structure model [Lan et al. 2012]	11.16%
DDCNA init	8.48%
DDCNA with learnt spatial filters	8.66%
DDCNA with learnt concepts	9.93%
DDCNA full	10.04
DDCNA full + pairwise terms	11.12%

Table A.3: Performance of our DDCNA approach that learns spatial concepts from data compared to the structured model of Lan et al. [2012]

Method	Extended relations & Structured Queries	Extended relations & Compositional Queries
DDCNA	7.90%	4.76%

Table A.4: Our approach on more challenging dataset: structured queries with the extended spatial relations, and compositional queries.

Extended set of spatial relations and compositional queries annotated by humans We extend our analysis to our new dataset that contains an extended set of spatial relations and complex, compositional queries. In particular, the annotations were obtained from human annotators and therefore the notion of spatial concepts has to be acquired in a learning-based approach. We show results of our full DDCNA approach on the simpler structured queries as well as the compositional ones in Table A.4. Note a drop in performance compared to the previous experiment as this is a more challenging setting. We achieve 7.90% mAP for the structured queries and 4.76% for the compositional ones. Furthermore, we show for the latter experiment all spatial concepts that we have learnt in Table A.2. They follow our intuition about the spatial layout as well as spatial spread of such spatial relations (e.g. ‘in” and ‘inside" are more focused). Table A.1 shows spatial relations that come from the dataset of Lan et al. [2012] containing only two spatial prepositions. Note that, the spatial templates learnt from the dataset of Lan et al. [2012] (shown in Table A.1) are less focused to the ones learnt from the extended set of spatial prepositions (shown in Table A.2). This confirms our hypothesis that we need datasets with a richer set of spatial prepositions. In addition to quantitative results, we also provide further visualizations of the retrieved images by our architecture given example queries. Figures starting from Figure 7.2 show the images together with their corresponding ranks sorted according to the confidence scores of the logistic regression. Further analysis revealed that most mistakes come from failure modes of the object detectors that our method is based on.

While our architecture has proven effective on this new type of challenging compositional queries, there is ample room for future work in order to close the performance gap to previous constraint settings of few spatial relations on simpler queries.

A.5 Qualitative results and Conclusions

In this document we have presented our novel approach to learn a joint representation of both images and language for image retrieval. Our DDCNA learns a representation of spatial relations only from example queries. Moreover, we have achieved results on par with previous work on structured queries under weaker modeling assumptions. The benefits of learnt spatial concepts is highlighted on a new, manually annotated, dataset of an extended set of spatial relations. Finally, we have shown that our architecture is capable of learning and predicting image relevance w.r.t. complex, compositional queries that have not previously been addressed. Our new dataset of human annotated queries enables this research direction. In this document we also provide further visualizations of the experiments with our architecture ('CNA full') on the extended and compositional queries. We show the top ranked retrieved images based on the score given by our architecture. We show images together with the corresponding ranking at the top of the image.

Figure A.2: Top ranked retrieved images from the query 'An airplane in front of a building'. We see a high recall achieved by our method and two clear mistakes - Rank 7 and Rank 15. Rank 7 is placed high in the ranking mainly due to false positive 'building' detection, and Rank 15 due to false positive 'airplane' detection.

Figure A.3: Top ranked retrieved images from the query 'Flowers in a vase'. Images Rank 4, 6, 7, 8, 9, 11, 12, ..., 15 are incorrectly ranked due to false positive 'vase' or 'flowers' detections with either strong signal response or large detection support.

Figure A.4: Top ranked retrieved images from the query 'Picture on the wall, above a bed'. We see a high recall achieved by our method. Although images Rank 2, Rank 6, Rank 8 and Rank 11 are mistakingly ranked high due to a strong false positive 'bed' detector, they are still reasonable. The architecture mistakingly ranks images Rank 4, 7, 9, 12 and 13 due to false positive 'bed' detection with either strong signal response or large detection support.

Figure A.5: Top ranked retrieved images from the query 'A van on the road below a window'. Images Rank 2, 4, 5, 7, 8, 9, 12, 13, 14 and 15 are clearly wrong. Interestingly the model hallucinates a 'van' (with strong signal response) and many 'windows' in the image Rank 12.

Figure A.6: Top ranked retrieved images from the query 'A chair in front of a door, on floor'. Images Rank 4, 6, 7, 10, 11, 14, 15 are placed incorrectly due to false positive detections.

Appendix B

Visual FactNet

Contents

This appendix supplements Chapters 9, 10, and 11 with a study, conducted in 2016, of effects of using an explicit scene representation together with a neural 'retrieving' mechanism for the Visual Turing Test. Concisely, the architecture on one hand uses explicit features extracted from detections as an image representations, and on the other hand it uses a neural-based approach to represent a question, and to decode an answer. Therefore, this architecture can be thought as a conceptual combination of the previous two architectures, a neural and logic-based ones, shown in the aforementioned chapters.

B.1 Introduction

Since our first instantiation of a Visual Turing Test [Malinowski and Fritz 2014a,b, 2015], also presented in the Chapters 8 and 9, we have seen a progress in the field ranging from a creation of new datasets to developing new methods. The latter can roughly be categorized according to the explicitness of a language or a visual representation as shown in Figure 5.3. Historically, first approaches towards a Visual Turing Challenge are symbolic based, presented in Chapter 9 [Malinowski and Fritz 2014a], and rely on a semantic parser. Such approaches require an explicit grammar from the language side, and object detectors or scene parsing on the visual side. The second generation of Visual Turing architectures are neural-based, and presented in Chapter 10. They use implicit representations: LSTM on the language side, and CNN on the visual side. Nowadays, we see an interest in attention-based architectures, or hybrid methods.

Finally, due to a recent development of Memory Networks [Sukhbaatar et al. 2015] for textual question answering, we can close the gap between symbolic and neural based approaches. Here, visual knowledge is collected from the image to form a database of "visual facts" (memories), and next an important information to an answer decoder is retrieved based on similarities between an encoded question and every memory cell that represent a

piece of visual knowledge about the image. From this perspective, originally used semantic parser has been replaced by a neural-based approach (LSTM), however, the method itself still allows for some degree of introspection.

B.2 Additional Analysis with Contemporary Architecture

In order to shed more light on the challenges inherent to answering questions on images, we pursue two directions. First, we propose a contemporary architecture that allows to inject ground truth information and thereby factoring out perception issues to a certain degree. Second, we propose a split of the DAQUAR dataset into different question types by which we can categorize the challenges into different groups.

B.2.1 Visual FactNet: Analyzing Question Answering by a Manipulable Memory Architecture

The approach proposed in Chapter 9 that is based on a symbolic reasoning has a great way to inspect and manipulate the visual representation and thereby gaining insights into the performance under different conditions. Yet, the latest models have shown great performance improvements by moving towards more implicit representations. For instance, neural-based approaches shown in Chapters 10 and 11 have doubled the performance compared to the prior symbolic approach – but at the cost of introspection, manipulable and interpretable representations. The latest memory networks as proposed by Sukhbaatar et al. [2015], strike a balance here. While still facilitating end-to-end learning, these approaches have an explicit memory representation. In fact, for the textual question answering, this memory is explicitly filled with factoids that are represented via a learnt embedding function. Similar architectures have been explored for VQA (e.g. Yang et al. [2015]).

Visual facts. Based on similar ideas, we present Visual FactNet, which we use as a tool for analyzing challenges in the 'question answering about images' task. The core idea is to encode ground truth visual information into the memory of a Memory Network in order to observe changes in performance under the "perfect vision". First, we represent the ground truth visual information as "visual facts" as shown in Figure B.1. Each object instance in the image is encoded in terms of one feature vector that includes the ground truth object class, size, color and location. All this information is represented as one-hot-vectors that are concatenated to form a memory entry for each instance. Continuous attributes like size and position are quantized in order to yield one-hot-vectors.

Architecture. Figure B.2 shows an overview of the proposed architecture that follows closely the Memory Network architecture of Sukhbaatar et al. [2015]. Therefore, we will only give an overview here. First, the question is encoded with a Long Short Term Memory [Hochreiter and Schmidhuber 1997] in order to yield a vector-based encoding. The Visual Facts are encoded into the memory via a matrix embedding A. The embedded question

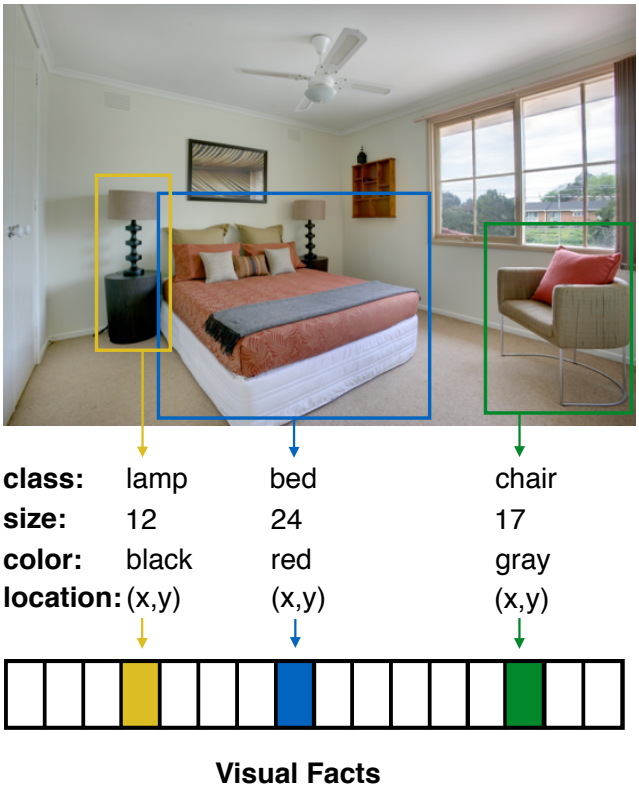


Figure B.1: Image to memory encoding.

is then used to retrieve relevant facts (by computing a scalar product between embedded memories and embedded question). The resulting vector is a sum of all embedded memories weighted by their relevance. Finally, this linear combination is added to the question encoding and an answer is determined by a dense layer and a consecutive softmax (only single word answers architecture is considered here[1]).

B.2.2 Performance Analysis by Question Type

In order to analyse the challenges of the task in more details, we have identified groups of question types in the DAQUAR dataset. We have defined the following question types:

- 133 questions involving colors
- 779 counting questions
- 160 questions involving size
- 2544 questions involving spatial relations
- 273 questions involving distances/proximity

[1]Note that we still evaluate the performance on multiple words answers on the test set.

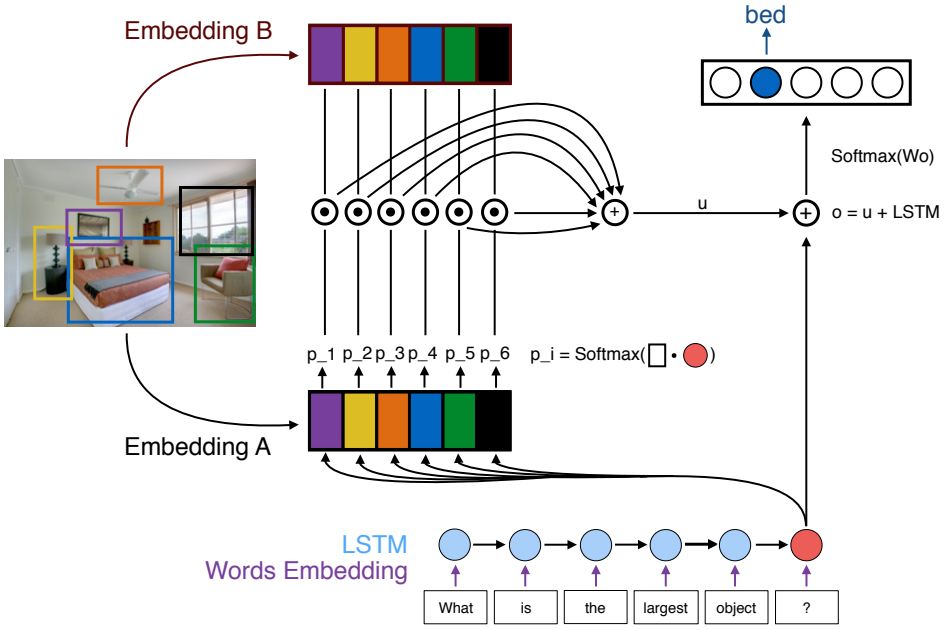


Figure B.2: Visual FactNet based on Memory Network architecture.

	Visual FactNet		Ask Your Neurons		Performance Difference	
	acc.	$WUPS_{0.9}$	acc.	$WUPS_{0.9}$	Δacc.	$\Delta WUPS_{0.9}$
all	24.31	29.81	19.43	25.28	4.88	4.5
color	23.31	30.95	24.06	34.53	-0.75	-3.58
count	32.73	38.55	28.75	34.93	3.98	3.62
size	48.12	53.00	16.77	23.98	31.35	29.02
spatial	20.64	26.24	14.62	20.65	6.02	5.59
distance	42.12	45.43	31.87	35.83	10.25	9.6

Table B.1: Performance evaluation on DAQUAR according to question types. For the sake of the visualization purpose, we only show results on two evaluation metrics.

In order to provide a contemporary baseline, we report results of our neural-based approach [Malinowski et al. 2015] and compare it to the Visual FactNet that has access to the ground truth visual information. We show results for both methods in Table B.1 as well as the performance difference. The strongest improvement is observed for questions involving size. We hypothesize that size estimates from ground truth bounding boxes are quite well aligned with human perception. Equally, we observe strong improvements for questions involving distance, which we attribute to a better reference resolution given the ground truth object information. The performance even degrades on the color questions when using the color estimates [Van De Weijer et al. 2007] in the Visual FactNet[2]. We hypothesize that this is due to an increased subjectivity for those questions and grounding of color names. These color subtle ties are also lost and cannot be recovered from if learning does

[2]Note that DAQUAR doesn't contain ground truth color information.

not start from the visual feature. Finally, the improvements in the Visual FactNet analysis are rather underwhelming for questions involving counting and spatial relations. Counting questions were already among the questions that are better answered and missing objects in the ground truth information plays a role here. The relatively small improvement on questions involving spatial relations points to the challenges of contextualization, reference frames and ambiguities that makes resolving such spatial language inherently difficult.

B.3 Summary

We present an architecture that 'sits' between two extreme approaches (Figure 5.3) – a symbolic based approach shown in Chapter 9 [Malinowski and Fritz 2014a], and a holistic, global frame neural architecture shown in Chapter 10 [Malinowski et al. 2015]. The architecture builds upon the prior work on Memory Networks [Sukhbaatar et al. 2015], and allows for some degree of the introspection. Our analysis shows that despite of injecting ground truth information, the results are far from being solved. We attribute this to the lack of joint training of the visual representation, inherent ambiguities, a mismatch between visual ground truth information and question-answering annotations, but also partially to issues in retrieving right information from the memories. At the same time, however, our analysis shows a big improvement in size and distance questions's types. Questions about counting or spatial relations also show an improvement but, arguably, a more thorough approach handling them should be developed.

Appendix C

Tutorial on Answering Questions about Images with Deep Learning

Contents

Together with the development of more accurate methods in Computer Vision and Natural Language Understanding, holistic architectures that answer on questions about the content of real-world images have emerged. In this tutorial, we build a neural-based approach to answer questions about images. We base our tutorial on two datasets: (mostly on) DAQUAR, and (a bit on) VQA. With small tweaks the models that we present here can achieve a competitive performance on both datasets, in fact, they are among the best methods that use a combination of LSTM with a global, full frame CNN representation of an image. We hope that after reading this tutorial, the reader will be able to use Deep Learning frameworks, such as Keras and introduced Kraino, to build various architectures that will lead to a further performance improvement on this challenging task.

C.1 Preface

In this tutorial[1] we build a few architectures that can answer questions about images. The architectures are based on our two papers on this topic: Malinowski et al. [2015] and Malinowski et al. [2016]; and more broadly, on our project towards a Visual Turing Test[2]. In particular, an encoder-decoder perspective of Malinowski et al. [2016] allows us to effectively experiment with various design choices. For the sake of simplicity, we only consider a classification-based approach to answer questions about images, although an approach that generate answers word-by-word is also studied in the community [Malinowski et al. 2015]. In the tutorial, we mainly focus on the DAQUAR dataset [Malinowski and Fritz 2014a], but a few possible directions to apply learnt techniques to VQA [Antol et al. 2015] are also pointed. First, we will get familiar with the task of answering questions about images, and a dataset that implements the task (due to a small size, we mainly use DAQUAR as it better serves an educational purpose that we aim at this tutorial). Next, we build a few blind models that answer questions about images without actually seeing such images. Such models already exhibit a reasonable performance as they can effectively learn various biases that exist in a dataset, which we also interpret as learning a common sense knowledge [Malinowski et al. 2015, 2016]. Subsequently, we build a few language+vision models that answer questions based on both a textual and a visual inputs. Finally, we leave the tutorial with a few possible research directions.

Technical aspects The tutorial is originally written using Python Notebook, which the reader is welcome to download[3] and use through the tutorial. Instructions necessary to run the Notebook version of this tutorial are provided in the following: `https://github.com/mateuszmalinowski/visual_turing_test-tutorial`. In this tutorial, we heavily use a Python code, and therefore it is expected the reader either already knows this language, or can quickly learn it. However, we made an effort to make this tutorial approachable to a wider audience. We use Kraino[3] that is a framework prepared for this tutorial in order to simplify the development of the question answering architectures. Under the hood, it uses Theano[4] [Bastien et al. 2012] and Keras[5] [Chollet 2015] – two frameworks to build Deep Learning models. We also use various CNNs representations extracted from images that can be downloaded as explained at the beginning of our Notebook tutorial[3]. We highlight exercises that a curious reader may attempt to solve in the following way.

```
This is an exercise.
```

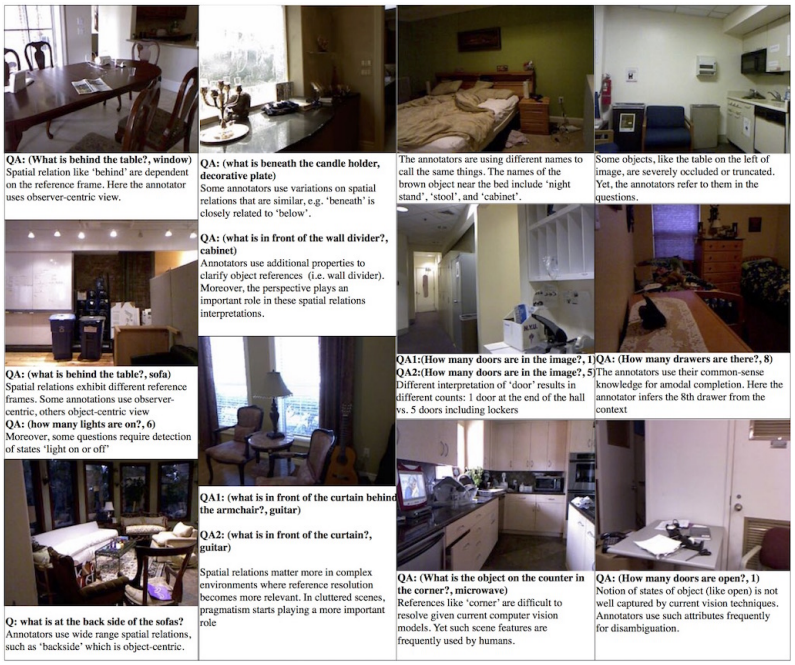


Figure C.1: Challenges present in the DAQUAR dataset.

C.2 Dataset

This section introduces the DAQUAR dataset [Malinowski and Fritz 2014a] from a programming perspective. Let us first list a few DAQUAR entries to become familiar with the format.

```
In [1]: ! head -15 data/daquar/qa.894.raw.train.format_triple

what is on the right side of the black telephone and on the left side of the red chair ?
desk
image3
what is in front of the white door on the left side of the desk ?
telephone
image3
what is on the desk ?
book, scissor, papers, tape_dispenser
image3
what is the largest brown objects ?
carton
```

[1] This tutorial was presented for the first time during the 2nd Summer School on Integrating Vision and Language: Deep Learning.

[2] http://mpii.de/visual_turing_test

[3] https://github.com/mateuszmalinowski/visual_turing_test-tutorial/blob/master/visual_turing_test.ipynb

[4] http://deeplearning.net/software/theano/

[5] https://keras.io

```
image3
what color is the chair in front of the white wall ?
red
image3
```

Note that the format is: question, answer (could be many answer words), and the image. Let us have a look at the Figure C.1. The figure lists images with associated question-answer pairs. It also comments on challenges associated with question-answer-image triplets. We see that to answer properly on the wide range of questions, an answerer not only needs to understand the scene visually or to just understand the question, but also, arguably, has to resort to the common sense knowledge, or even know the preferences of the person asking the question, e.g. what 'behind' exactly means in 'What is behind the table?'. Hence, architectures that answer questions about images have to face many challenges. Ambiguities make it also difficult to judge the provided answers. We revisit this issue in a later section. Meantime, a curious reader may try to answer the following question.

```
Can you spot ambiguities that are present in the first column of the figure?
Think of a spatial relationship between an observer, object of interest, and the world.
```

The following code returns a dictionary of three views on the DAQUAR dataset. For now, we look only into the 'text' view. dp['text'] returns a function from a dataset split into the dataset's textual view. Executing the following code makes it more clear.

```
In [ ]: #TODO: Execute the following procedure (Shift+Enter in the Notebook)
        from kraino.utils import data_provider

        dp = data_provider.select['daquar-triples']
        train_text_representation = dp['text'](train_or_test='train')
```

This view specifies how questions are ended ('?'), answers are ended ('.'), answer words are delimited (DAQUAR sometimes has a set of answer words as an answer, for instance 'knife, fork' may be an answer answer), but most important, it has questions (key 'x'), answers (key 'y'), and names of the corresponding images (key 'img_name').

```
In [ ]: # let us check some entries of the text's representation
        n_elements = 10
        print('== Questions:')
        print_list(train_text_representation['x'][:n_elements])
        print
        print('== Answers:')
        print_list(train_text_representation['y'][:n_elements])
        print
        print('== Image Names:')
        print_list(train_text_representation['img_name'][:n_elements])
```

Summary DAQUAR consists of question-answer-image triplets. Question-answer pairs for different folds are accessible from executing the following code.

```
data_provider.select['text']
```

Finally, as we see in Figure C.1, DAQUAR poses many challenges in designing good architectures, or evaluation metrics.

C.3 Textual Features

We have an access to a textual representation of questions. This is however not very helpful since neural networks expect a numerical input, and hence we cannot really work with the raw text. We need to transform the textual input into some numerical value or a vector of values. One particularly successful representation is called one-hot vector and it is a binary vector with exactly one non-zero entry. This entry points to the corresponding word in the vocabulary. See the illustration shown in Figure C.2.

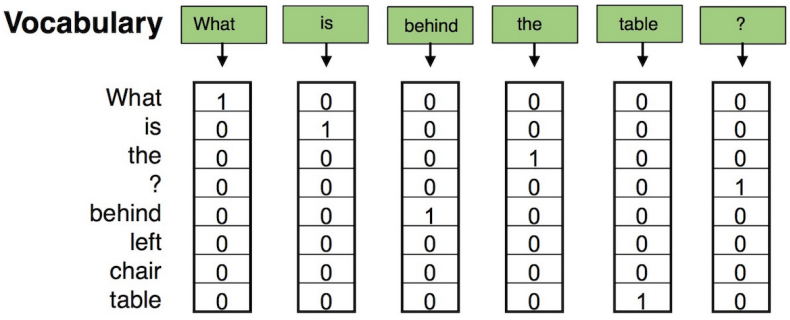


Figure C.2: One hot representations of the textual words in the question.

The reader can pause here a bit to answer the following questions.

```
Can you sum up the one-hot vectors for the 'What table is behind the table?'.
How would you interpret the resulting vector?
Why is it a good idea to work with one-hot vector represetantions of the text?
```

As we see from the illustrative example above, we first need to build a suitable vocabulary from our raw textual training data, and next transform them into one-hot representations. The following code can do this.

```
In [ ]: from toolz import frequencies
        train_raw_x = train_text_representation['x']
        # we start from building the frequencies table
        wordcount_x = frequencies(' '.join(train_raw_x).split(' '))
        # print the most and least frequent words
        n_show = 5
        print(sorted(wordcount_x.items(), key=lambda x: x[1], reverse=True)[:n_show])
        print(sorted(wordcount_x.items(), key=lambda x: x[1])[:n_show])
```

In many parts of this tutorial, we use Kraino, which was developed for the purpose of this tutorial to simplify the development of various 'question answering' models through prototyping.

```
from kraino.utils.input_output_space import build_vocabulary

# This function takes wordcounts,
# and returns word2index - mapping from words into indices,
# and index2word - mapping from indices to words.
word2index_x, index2word_x = build_vocabulary(
    this_wordcount=wordcount_x,
    truncate_to_most_frequent=0)
word2index_x
```

In addition, we use a few special, extra symbols that do not occur in the training dataset. Most important are $< pad >$ and $< unk >$. We use the former to pad sequences in order to have the same number of temporal elements; we use the latter for words (at test time) that do not exist in the training set. Armed with the vocabulary, we can build one-hot representations of the training data. However, this is not necessary and may even be wasteful. Our one-hot representation of the input text does not explicitly build long and sparse vectors, but instead it operates on indices. The example from Figure C.2 would be encoded as [0,1,4,2,7,3].

Due to the sparsity existing in the one-hot representation, we can more efficiently operate on indices instead of performing full linear transformations by matrix-vector multiplications. This is reflected in the following claim.

Claim: Let x be a binary vector with exactly one value 1 at the position $index$, that is $x[index] = 1$. Then

$$W[:, index] = Wx$$

where $W[:, b]$ denotes a vector built from a column b of W. This shows that matrix-vector multiplication can be replaced by retrieving a right vector of parameters according to the index.

`Can you show that the claim is valid?`

We can encode textual questions into one-hot vector representations by executing the following code.

```
In [ ]: from kraino.utils.input_output_space import encode_questions_index
        one_hot_x = encode_questions_index(train_raw_x, word2index_x)
        print(train_raw_x[:3])
        print(one_hot_x[:3])
```

As we can see, the sequences have different number of elements. We can pad the sequences to have the same length by setting up $MAXLEN$.

```
from keras.preprocessing import sequence
MAXLEN=30
train_x = sequence.pad_sequences(one_hot_x, maxlen=MAXLEN)
train_x[:3]
```

We do the same with the answers.

```
In [ ]: # for simplicity, we consider only first answer words;
In [ ]: # that is, if answer is 'knife,fork' we encode only 'knife'
MAX_ANSWER_TIME_STEPS=1

from kraino.utils.input_output_space import encode_answers_one_hot
train_raw_y = train_text_representation['y']
wordcount_y = frequencies(' '.join(train_raw_y).split(' '))
word2index_y, index2word_y = build_vocabulary(this_wordcount=wordcount_y)
train_y, _ = encode_answers_one_hot(
    train_raw_y,
    word2index_y,
    answer_words_delimiter=train_text_representation['answer_words_delimiter'],
    is_only_first_answer_word=True,
    max_answer_time_steps=MAX_ANSWER_TIME_STEPS)
print(train_x.shape)
print(train_y.shape)
```

At the last step, we encode test questions. We need them later to see how well our models generalize to new question-answer-image triplets. Remember, however, that we should use the vocabulary we generated from the training samples.

`Why should we use the training vocabulary to encode test questions?`

```
In [ ]: test_text_representation = dp['text'](train_or_test='test')
test_raw_x = test_text_representation['x']
test_one_hot_x = encode_questions_index(test_raw_x, word2index_x)
test_x = sequence.pad_sequences(test_one_hot_x, maxlen=MAXLEN)
print_list(test_raw_x[:3])
test_x[:3]
```

With the encoded question-answer pairs we finish this section. However, before delving into details of building and training new models, let us have a look at the summary to see bigger picture.

Summary We started from raw questions from the training set. We use them to build a vocabulary. Next, we encode questions into sequences of one-hot vectors based on the vocabulary. Finally, we use the same vocabulary to encode questions from the test set. If a word is absent, we use an extra token $< unk >$ to denote this fact, so that we encode the $< unk >$ token, not the word.

C.4 Language Only Models

C.4.0.1 Training

As you may already know, we train models by weights updates. Let x and y be training samples (an input, and an output), and $\ell(x, y)$ be an objective function. The formula for weights updates is:

$$w := w - \alpha \nabla \ell(x, y; w)$$

with α that we call the learning rate, and ∇ that is a gradient wrt. the weights w. The learning rate is a hyper-parameter that must be set in advance. The rule shown above is called the SGD update, but other variants are also possible. In fact, we use its variant called ADAM [Kingma and Ba 2014].

We cast the question answering problem into a classification framework, so that we classify an input x into some class that represents an answer word. Therefore, we use, commonly used in the classification, logistic regression as the objective:

$$\ell(x, y; w) := \sum_{y' \in \mathcal{C}} \mathbb{1}\{y' = y\} \log p(y' \mid x, w)$$

where $\mathcal{C}$ is a set of all classes, and $p(y \mid x, w)$ is the softmax: $e^{w^y \phi(x)} / \sum_z e^{w^z \phi(x)}$. Here $\phi(x)$ denotes an output of a model (more precisely, it is often a response of a neural network to the input, just before softmax of the neural network is applied). Note, however, that another variant of providing answers, called the answer generation, is also possible [Malinowski et al. 2015]. For training, we need to execute the following code.

```
training(gradient_of_the_model, optimizer='Adam')
```

Summary Given a model, and an optimization procedure (SGD, Adam, etc.) all we need is to compute gradient of the model $\nabla \ell(x, y; w)$ wrt. to its parameters w, and next plug it to the optimization procedure.

C.4.0.2 Theano

Since computing gradients $\nabla \ell(x, y; w)$ may quickly become tedious, especially for more complex models, we search for tools that could automatize this process. Imagine that you build a model M and you get its gradient ∇M by just executing the tool, something like the following piece of code.

```
nabla_M = compute_gradient_symbolically(M,x,y)
```

This would definitely speed up prototyping. Theano [Bastien et al. 2012] is such a tool that is specifically tailored to work with deep learning models. For a broader understanding of Theano, you can check a suitable tutorial[6].

The following coding example defines ReLU, a popular activation function defined as $ReLU(x) = \max(x, 0)$, as well as derive its derivative using Theano. Note however that, with this example, we obviously only scratch the surface.

[6] For instance, http://deeplearning.net/tutorial/.

```
In [ ]: import theano
        import theano.tensor as T

        # Theano uses symbolic calculations,
                # so we need to first create symbolic variables
        theano_x = T.scalar()
        # we define a relationship between a symbolic input and a symbolic output
        theano_y = T.maximum(0,theano_x)
        # now it's time for a symbolic gradient wrt. to symbolic variable x
        theano_nabla_y = T.grad(theano_y, theano_x)

        # we can see that both variables are symbolic, they don't have any numerical values
        print(theano_x)
        print(theano_y)
        print(theano_nabla_y)

        # theano.function compiles the symbolic representation of the network
        theano_f_x = theano.function([theano_x], theano_y)
        print(theano_f_x(3))
        print(theano_f_x(-3))
        # and now for gradients

        nabla_f_x = theano.function([theano_x], theano_nabla_y)
        print(nabla_f_x(3))
        print(nabla_f_x(-3))
```

Can you derive a derivative of ReLU on your own? Consider two cases.

It should also be mentioned that ReLU is a non-differentiable function at the point 0, and therefore, technically, we compute its sub-gradient – this is however still fine for Theano.

Summary To compute gradient symbolically, we can use Theano. This speeds up prototyping, and hence developing new question answering models.

C.4.0.3 Keras

Keras [Chollet 2015] builds upon Theano, and significantly simplifies creating new deep learning models as well as training such models, effectively speeding up the prototyping even further. Keras also abstracts away from some technical burden such as a symbolic variable creation. Many examples of using Keras can be found by following the links: https://keras.io/getting-started/sequential-model-guide/, and https://keras.io/getting-started/functional-api-guide/. Note that, in the tutorial we use an older sequential model. Please also pay attention to the version of the Keras, since not all versions are compatible with this tutorial.

C.4.0.4 Models

For the purpose of the Visual Turing Test, and this tutorial, we have compiled a light framework that builds on top of Keras, and simplify building and training 'question answering' machines. With the tradition of using Greek names, we call it Kraino. Note that some parts of the Kraino, such as a data provider, were already covered in this tutorial.

In the following, we will go through BOW and LSTM approaches to answer questions about images, but, surprisingly, without the images. It turns out that a substantial fraction of questions can be answered without an access to an image, but rather by resorting to a common sense (or statistics of the dataset). For instance, 'what can be placed at the table?', or 'How many eyes this human have?'. Answers like 'chair' and '2' are quite likely to be good answers.

C.4.0.5 BOW

Figure C.3 illustrates the BOW (Bag Of Words) method. As we have already seen before, we first encode the input sentence into one-hot vector representations. Such a (very) sparse representation is next embedded into a denser space by a matrix W_e. Next, the denser representations are summed up and classified via 'Softmax'. Notice that, if W_e were an identity matrix, we would obtain a histogram of the word's occurrences.

```
What is your biggest complain about such a BOW representation?
What happens if instead of 'What is behind the table' we would have
'is What the behind table'? How does the BOW representation change?
```

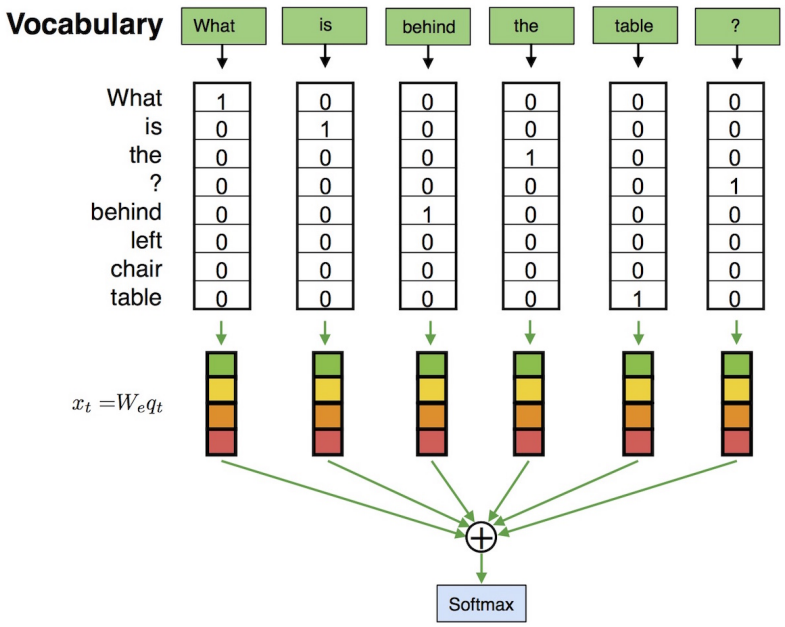


Figure C.3: Bag-Of-Words (BOW) representation of the input that is next follow by 'Softmax'.

Let us now define a BOW model using our tools.

In [ ]:
```
#== Model definition

        # First we define a model using keras/kraino
        from keras.layers.core import Activation
        from keras.layers.core import Dense
        from keras.layers.core import Dropout
        from keras.layers.core import TimeDistributedMerge
        from keras.layers.embeddings import Embedding

        from kraino.core.model_zoo import AbstractSequentialModel
        from kraino.core.model_zoo import AbstractSingleAnswer
        from kraino.core.model_zoo import AbstractSequentialMultiplewordAnswer
        from kraino.core.model_zoo import Config
        from kraino.core.keras_extensions import DropMask
        from kraino.core.keras_extensions import LambdaWithMask
        from kraino.core.keras_extensions import time_distributed_masked_ave

        # This model inherits from AbstractSingleAnswer,
        # and so it produces single answer words
        # To use multiple answer words,
        # you need to inherit from AbstractSequentialMultiplewordAnswer
        class BlindBOW(AbstractSequentialModel, AbstractSingleAnswer):
            """
            BOW Language only model that produces single word answers.
            """
            def create(self):
                self.add(Embedding(
                        self._config.input_dim,
                        self._config.textual_embedding_dim,
                        mask_zero=True))
                self.add(LambdaWithMask
                        (time_distributed_masked_ave, output_shape=[self.output_shape[2]]))
                self.add(DropMask())
                self.add(Dropout(0.5))
                self.add(Dense(self._config.output_dim))
                self.add(Activation('softmax'))


In [ ]: model_config = Config(
            textual_embedding_dim=500,
            input_dim=len(word2index_x.keys()),
            output_dim=len(word2index_y.keys()))
        model = BlindBOW(model_config)
        model.create()
```

```
        model.compile(
            loss='categorical_crossentropy',
            optimizer='adam')
        text_bow_model = model
```

```
In [ ]: #== Model training
        text_bow_model.fit(
            train_x,
            train_y,
            batch_size=512,
            nb_epoch=40,
            validation_split=0.1,
            show_accuracy=True)
```

C.4.0.6 Recurrent Neural Network

Although BOW is working pretty well, there is still something very disturbing about this approach. Consider the following question: 'what is on the right side of the black telephone and on the left side of the red chair ?' If we swap 'chair' with 'telephone' in the question, we would get a different meaning. Recurrent Neural Networks (RNNs) have been developed to mitigate this issue by directly processing time series. As Figure C.4 illustrates, the (temporarily) first word embedding is given to an RNN unit. The RNN unit next processes such an embedding and outputs to the second RNN unit. This unit takes both the output of the first RNN unit and the 2nd word embedding as inputs, and outputs some algebraic combination of both inputs. And so on. The last recurrent unit builds the representation of the whole sequence. Its output is next given to Softmax for the classification. One among the challenges that such approaches have to deal with is maintaining long-term dependencies. Roughly speaking, as new inputs are coming in the following steps it is getting easier to 'forget' information from the beginning (the first temporal step). LSTM [Hochreiter and Schmidhuber 1997] and GRU [Cho et al. 2014] are two particularly popular Recurrent Neural Networks that can preserve such longer dependencies to some extent[7].
Let us create a Recurrent Neural Network in the following.

```
In [ ]: #== Model definition

        # First we define a model using keras/kraino
        from keras.layers.core import Activation
        from keras.layers.core import Dense
        from keras.layers.core import Dropout
        from keras.layers.embeddings import Embedding
        from keras.layers.recurrent import GRU
        from keras.layers.recurrent import LSTM
```

[7] http://karpathy.github.io/2015/05/21/rnn-effectiveness/

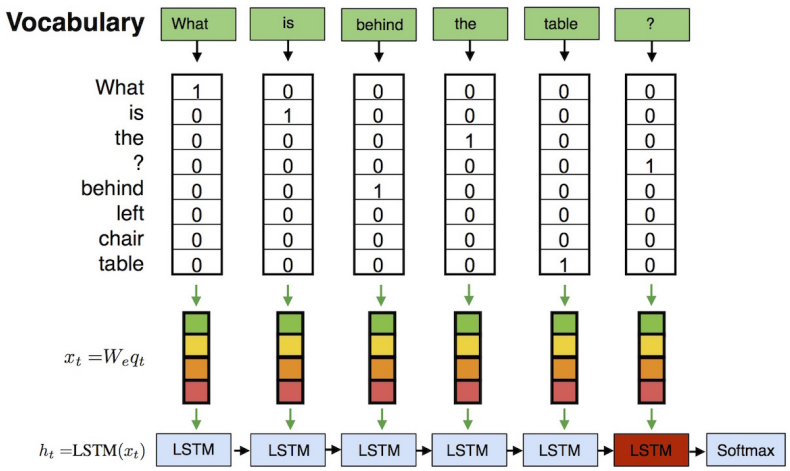


Figure C.4: Recurrent Neural Network.

```
from kraino.core.model_zoo import AbstractSequentialModel
from kraino.core.model_zoo import AbstractSingleAnswer
from kraino.core.model_zoo import AbstractSequentialMultiplewordAnswer
from kraino.core.model_zoo import Config
from kraino.core.keras_extensions import DropMask
from kraino.core.keras_extensions import LambdaWithMask
from kraino.core.keras_extensions import time_distributed_masked_ave

# This model inherits from AbstractSingleAnswer,
# and so it produces single answer words
# To use multiple answer words,
# you need to inherit from AbstractSequentialMultiplewordAnswer
class BlindRNN(AbstractSequentialModel, AbstractSingleAnswer):
    """
    RNN Language only model that produces single word answers.
    """
    def create(self):
        self.add(Embedding(
                self._config.input_dim,
                self._config.textual_embedding_dim,
                mask_zero=True))
        #TODO: Replace averaging with RNN (you can choose between LSTM and GRU)
        self.add(GRU(self._config.hidden_state_dim,
                     return_sequences=False))
        self.add(Dropout(0.5))
        self.add(Dense(self._config.output_dim))
        self.add(Activation('softmax'))
```

```
In [ ]: model_config = Config(
            textual_embedding_dim=500,
            hidden_state_dim=500,
            input_dim=len(word2index_x.keys()),
            output_dim=len(word2index_y.keys()))
        model = BlindRNN(model_config)
        model.create()
        model.compile(
            loss='categorical_crossentropy',
            optimizer='adam')
        text_rnn_model = model

In [ ]: #== Model training
        text_rnn_model.fit(
            train_x,
            train_y,
            batch_size=512,
            nb_epoch=40,
            validation_split=0.1,
            show_accuracy=True)
```

The curious reader is encouraged to experiment with the language-only models. For instance, to see the influence of particular modules to the overall performance, the reader can do the following exercise.

```
Change the number of hidden states.
Change the number of epochs used to train a model.
Modify models by using more RNN layers, or deeper classifiers.
```

Summary RNN models, as opposite to BOW, consider order of the words in the question. Moreover, a substantial number of questions can be answered without any access to images. This can be explained as models learn some specific dataset statistics, some of them can be interpreted as common sense knowledge.

C.5 Evaluation Measures

To be able to monitor a progress on a task, we need to find ways to evaluate architectures on the task. Otherwise, we would not know how to judge architectures, or even worse, we would not even know what the goal is. Moreover, we should also aim at automatic evaluation measures, otherwise reproducibility is questionable, and the evaluation costs are high.

C.5.0.1 Ambiguities

Although an early work on the Visual Turing Test argues for keeping the answer words from a fixed vocabulary in order to keep an evaluation simpler [Malinowski and Fritz 2014a,b, 2015], it is still difficult to automatically evaluate architectures due to ambiguities that occur in the answers. We have ambiguities in naming objects, sometimes due to synonyms, but sometimes due to fuzziness. For instance, is 'chair' == 'armchair' or 'chair' != 'armchair' or something in between? Such semantic boundaries become even more fuzzy when we increase the number of categories. We could easily find a mutually exclusive set of 10 different categories, but what if there are 1000 categories, or 10000 categories? Arguably, we cannot think in terms of an equivalence class anymore, but rather in terms of similarities. That is 'chair' is semantically more similar to 'armchair', than to 'horse'. This simple example shows the main drawback of a traditional, binary evaluation measure which is Accuracy. This metric scores 1 if the names are the same and 0 otherwise. So that Acc('chair', 'armchair') == Acc('chair', 'horse'). We call these ambiguities, word-level ambiguities, but there are other ambiguities that are arguably more difficult to handle. For instance, the same question can be phrased in multiple other ways. The language of spatial relations is also ambiguous. Language tends to be also rather vague - we sometimes skip details and resort to common sense. Some ambiguities are rooted in a culture. To address world-level ambiguities, Malinowski and Fritz [2014a] propose WUPS. To address ambiguities caused by various interpretations of an image or a question, Malinowski et al. [2015] propose Consensus measures. For the sake of simplicity, in this tutorial, we only use WUPS. On the other hand, arguably, it is easier to evaluate architectures on DAQUAR than on Image Captioning datasets. The former restricts the output space to N categories, while it still requires a holistic comprehension. Let us remind that Figure C.1 shows a few ambiguities that exists in DAQUAR.

C.5.0.2 Wu-Palmer Similarity

Given an ontology a Wu-Palmer Similarity between two words (or broader concepts) is a soft measure defined as

$$WuP(a,b) := \frac{lca(a,b)}{depth(a) + depth(b)}$$

where $lca(a,b)$ is the least common ancestor of a and b, and $depth(a)$ is depth of a in the ontology. Figure C.5 shows a toy-sized ontology. The curious reader can, based on Figure C.5, address the following questions.

```
What is WuP(Dog, Horse) and WuP(Dog, Dalmatian) according to the toy-sized ontology?
Can you  calculate Acc(Dog, Horse) and Acc(Dog, Dalmatian)?
```

C.5.0.3 WUPS

Wu-Palmer Similarity depends on the choice of ontology. One popular, large ontology is WordNet [Miller 1995; Fellbaum 1999]. Although Wu-Palmer Similarity may work on shallow ontologies, we are rather interested in ontologies with hundreds or even thousands of categories. In indoor scenarios, it turns out that many indoor 'things' share similar levels

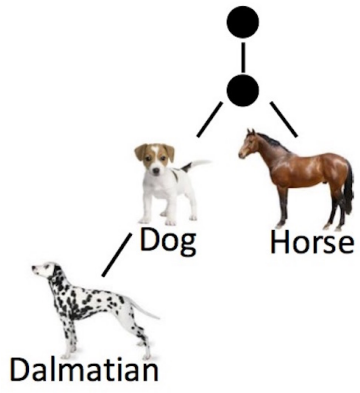


Figure C.5: A toy-sized ontology.

in the ontology, and hence Wu-Palmer Similarities are very small between two entities. The following code exemplifies the issue.

```
In [ ]: from nltk.corpus import wordnet as wn
        armchair_synset = wn.synset('armchair.n.01')
        chair_synset = wn.synset('chair.n.01')
        wardrobe_synset = wn.synset('wardrobe.n.01')

        print(armchair_synset.wup_similarity(armchair_synset))
        print(armchair_synset.wup_similarity(chair_synset))
        print(armchair_synset.wup_similarity(wardrobe_synset))
        wn.synset('chair.n.01').wup_similarity(wn.synset('person.n.01'))
```

As we can see that 'armchair' and 'wardrobe' are surprisingly close to each other. It is because, for large ontologies, all the indoor 'things' are semantically 'indoor things'. This issue has motivated us to define thresholded Wu-Palmer Similarity Score, defined as follows

$$\begin{array}{rl} WuP(a,b) & \text{if } WuP(a,b) \geq \tau \\ 0.1 \cdot WuP(a,b) & \text{otherwise} \end{array}$$

where τ is a hand-chosen threshold. Empirically, we found that $\tau = 0.9$ works fine on DAQUAR [Malinowski and Fritz 2014a]. Moreover, since DAQUAR has answers as sets of answer words, so that 'knife,fork' == 'fork,knife', we have extended the above measure to work with the sets. We call it Wu-Palmer Set score, or shortly WUPS.

A detailed exposition of WUPS is beyond this tutorial, but a curious reader is encouraged to read the 'Performance Measure' paragraph in Malinowski and Fritz [2014a]. Note that the measure in Malinowski and Fritz [2014a] is defined broader, and essentially it abstracts away from any particular similarities such as Wu-Palmer Similarity, or an ontology. WUPS at 0.9 is WUPS with threshold $\tau = 0.9$. It is worth noting, that a practical implementation of WUPS needs to deal with synsets. Thus it is recommended to download the script from http://datasets.d2.mpi-inf.mpg.de/mateusz14visual-turing/calculate_wups.py or re-implement it with caution.

C.5.0.4 Consensus

The consensus measure handles ambiguities that are caused by various interpretations of a question or an image. In this tutorial, we do not cover this measure. A curious reader is encouraged to read the 'Human Consensus' in Malinowski et al. [2015].

C.5.0.5 A few caveats

We present a few caveats when using WUPS. These can be especially useful if one wants to adapt WUPS to other datasets.

Lack of coverage Since WUPS is based on an ontology, not always it recognizes words. For instance 'garbage bin' is missing, but 'garbage can' is perfectly fine. You can check it by yourself, either with the source code provided above, or by using an online script[8].

Synsets The execution of the following code

```
wn.synsets('chair')
```

produces a list with many elements. These elements are semantically equivalent[9].

For instance the following definition of 'chair'

```
wn.synset('chair.n.03').definition()
```

indicates a person (e.g. a chairman). Indeed, the following gives quite high value

```
wn.synset('chair.n.03').wup_similarity(wn.synset('person.n.01'))
```

however the following one has a more preferred, much lower value

```
wn.synset('chair.n.01').wup_similarity(wn.synset('person.n.01'))
```

How to deal with such a problem? In DAQUAR we take an optimistic perspective and always consider the highest similarity score. This works with WUPS 0.9 and a restricted indoor domain with a vocabulary based only on the training set. To sum up, this issue should be taken with a caution whenever WUPS is adapted to other domains.

Ontology Since WUPS is based on an ontology, specifically on WordNet, it may give different scores on different ontologies, or even on different versions of the same ontology.

Threshold A good threshold τ is dataset dependent. In our case $\tau = 0.9$ seems to work well, while $\tau = 0.0$ is too forgivable and is rather reported due to the 'historical' reasons. However, following our papers, you should still consider to report plain set-based accuracy scores (so that Acc('knife,'fork','fork,knife')==1; it can be computed by our script[10] using the argument -1 to WUPS.

[8] http://wordnetweb.princeton.edu/perl/webwn
[9] https://en.wikipedia.org/wiki/Synonym_ring
[10] http://datasets.d2.mpi-inf.mpg.de/mateusz14visual-turing/calculate_wups.py

C.5.0.6 Summary

WUPS is an evaluation measure that works with sets and word-level ambiguities. Arguably, WUPS at 0.9 is the most practical measure.

C.6 New Predictions

Once the training of our models is over, we can evaluate their performance on a previously unknown test set. In the following, we show how to make predictions using the already discussed blind models.

C.6.0.1 Predictions - BOW

We start from encoding textual input into one-hot vector representations.

```
In [ ]: test_text_representation = dp['text'](train_or_test='test')
        test_raw_x = test_text_representation['x']
        test_one_hot_x = encode_questions_index(test_raw_x, word2index_x)
        test_x = sequence.pad_sequences(test_one_hot_x, maxlen=MAXLEN)
```

Given encoded test questions, we use the maximum likelihood principle to withdraw answers.

```
In [ ]: from numpy import argmax
        # predict the probabilities for every word
        predictions_scores = text_bow_model.predict([test_x])
        print(predictions_scores.shape)
        # follow the maximum likelihood principle, and get the best indices to vocabulary
        predictions_best = argmax(predictions_scores, axis=-1)
        print(predictions_best.shape)
        # decode the predicted indices into word answers
        predictions_answers = [index2word_y[x] for x in predictions_best]
        print(len(predictions_answers))
```

Now, we evaluate the answers using WUPS.

```
In [ ]: from kraino.utils import print_metrics
        test_raw_y = test_text_representation['y']
        _ = print_metrics.select['wups'](
                gt_list=test_raw_y,
                pred_list=predictions_answers,
                verbose=1,
                extra_vars=None)
```

Let us see the predictions.

```
In [ ]: from numpy import random
        test_image_name_list = test_text_representation['img_name']
```

```
indices_to_see = random.randint(low=0, high=len(test_image_name_list), size=5)
for index_now in indices_to_see:
    print(test_raw_x[index_now], predictions_answers[index_now])
```

Without looking at images, a curious reader may attempt to answer the following questions.

```
Do you agree with the answers given above? What are your guesses?
Of course, neither you nor the model have seen any images so far.
```

However, what happens if the reader can actually see the images?

```
Execute the code below.
Do your answers change after seeing the images?
```

```
In [1]: from matplotlib.pyplot import axis
        from matplotlib.pyplot import figure
        from matplotlib.pyplot import imshow

        import numpy as np
        from PIL import Image

        %matplotlib inline
        for index_now in indices_to_see:
            image_name_now = test_image_name_list[index_now]
            pil_im = Image.open('data/daquar/images/{0}.png'.format(image_name_now), 'r')
            fig = figure()
            fig.text(.2,.05,test_raw_x[index_now], fontsize=14)
            axis('off')
            imshow(np.asarray(pil_im))
```

Finally, let us also see the ground truth answers by executing the following code.

```
In [ ]: print('question, prediction, ground truth answer')
        for index_now in indices_to_see:
            print(test_raw_x[index_now],
                  predictions_answers[index_now], test_raw_y[index_now])
```

In the code above, we have randomly taken questions, and hence different executions of the code may lead to different answers.

C.6.0.2 Predictions - RNN

Let us do similar predictions with a Recurrent Neural Network. This time, we use Kraino, to make the code shorter.

```
In [ ]: from kraino.core.model_zoo import word_generator
        # we first need to add word_generator to _config
```

```
        # (we could have done this before, in the Config constructor)
        # we use maximum likelihood as a word generator
        text_rnn_model._config.word_generator = word_generator['max_likelihood']
        predictions_answers = text_rnn_model.decode_predictions(
            X=test_x,
            temperature=None,
            index2word=index2word_y,
            verbose=0)
```

```
In [ ]: _ = print_metrics.select['wups'](
                gt_list=test_raw_y,
                pred_list=predictions_answers,
                verbose=1,
                extra_vars=None)
```

A curious reader is encouraged to try the following exercise.

`Visualise question, predicted answers, ground truth answers as before. Check also images.`

C.7 Visual Features

All the considered so far architectures predict answers based only on questions, even though the questions concern images. Therefore, in this section, we also build visual features. As shown in Figure C.6, a quite common practice is to:

1. Take an already pre-trained CNN; often pre-training is done in some large-scale classification task such as ImageNet [Russakovsky et al. 2014].

2. 'Chop off' a CNN representation after some layer. We use responses of that layer as visual features.

In this tutorial, we use features extracted from the second last 4096 dimensional layer of the VGG Net [Simonyan and Zisserman 2015]. We have already extracted features in advance using Caffe [Jia et al. 2014] - another excellent framework for deep learning, particularly good for CNNs.
The following code gives visual features aligned with textual features.

```
In [ ]: # this contains a list of the image names of our interest;
        # it also makes sure that visual and textual features are aligned correspondingly
        train_image_names = train_text_representation['img_name']
        # the name for visual features that we use
        # CNN_NAME='vgg_net'
        # CNN_NAME='googlenet'
        CNN_NAME='fb_resnet'
        # the layer in CNN that is used to extract features
```

Figure C.6: Features extractor. Neural responses of some layer to the visual input are considered as features.

```
# PERCEPTION_LAYER='fc7'
# PERCEPTION_LAYER='pool5-7x7_s1'
# PERCEPTION_LAYER='res5c-152'
# l2 prefix since there are l2-normalized visual features
PERCEPTION_LAYER='l2_res5c-152'

train_visual_features = dp['perception'](
    train_or_test='train',
    names_list=train_image_names,
    parts_extractor=None,
    max_parts=None,
    perception=CNN_NAME,
    layer=PERCEPTION_LAYER,
    second_layer=None
    )
train_visual_features.shape
```

C.8 Vision+Language

Given visual features, we can now build a full model that answer questions about images. As we can see in Figure C.1, it is hard to answer correctly on questions without seeing images. Let us create an input as a pair of textual and visual features using the following code.

```
In [ ]: train_input = [train_x, train_visual_features]
```

In the following, we investigate two approaches to question answering: an orderless BOW, and an RNN.

C.8.0.1 BOW + Vision

Similarly to our blind model, we start with a BOW encoding of a question. Here, we explore two ways of combining both modalities (circle with 'C' in Figure C.7): concatenation, and piece-wise multiplication. For the sake of simplicity, we do not fine-tune the

visual representation (dotted line symbolizes the barrier that blocks back-propagation in Figure C.7).

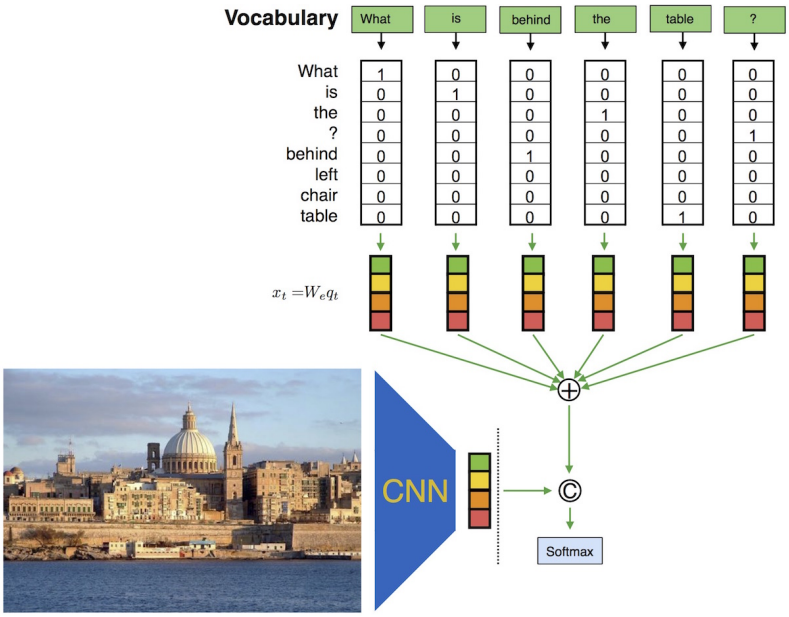


Figure C.7: BOW with visual features.

```
In [ ]: #== Model definition

        # First we define a model using keras/kraino
        from keras.models import Sequential
        from keras.layers.core import Activation
        from keras.layers.core import Dense
        from keras.layers.core import Dropout
        from keras.layers.core import Layer
        from keras.layers.core import Merge
        from keras.layers.core import TimeDistributedMerge
        from keras.layers.embeddings import Embedding

        from kraino.core.model_zoo import AbstractSequentialModel
        from kraino.core.model_zoo import AbstractSingleAnswer
        from kraino.core.model_zoo import AbstractSequentialMultiplewordAnswer
        from kraino.core.model_zoo import Config
        from kraino.core.keras_extensions import DropMask
        from kraino.core.keras_extensions import LambdaWithMask
        from kraino.core.keras_extensions import time_distributed_masked_ave

        # This model inherits from AbstractSingleAnswer,
        # and so it produces single answer words
```

```
        # To use multiple answer words,
        # you need to inherit from AbstractSequentialMultiplewordAnswer
        class VisionLanguageBOW(AbstractSequentialModel, AbstractSingleAnswer):
            """
            BOW Language only model that produces single word answers.
            """
            def create(self):
                language_model = Sequential()
                language_model.add(Embedding(
                        self._config.input_dim,
                        self._config.textual_embedding_dim,
                        mask_zero=True))
                language_model.add(LambdaWithMask(
                        time_distributed_masked_ave,
                        output_shape=[language_model.output_shape[2]]))
                language_model.add(DropMask())
                visual_model = Sequential()
                if self._config.visual_embedding_dim > 0:
                    visual_model.add(Dense(
                            self._config.visual_embedding_dim,
                            input_shape=(self._config.visual_dim,)))
                else:
                    visual_model.add(Layer(input_shape=(self._config.visual_dim,)))
                self.add(Merge([language_model,
                        visual_model], mode=self._config.multimodal_merge_mode))
                self.add(Dropout(0.5))
                self.add(Dense(self._config.output_dim))
                self.add(Activation('softmax'))
```

In [ ]:
```
        # dimensionality of embeddings
        EMBEDDING_DIM = 500
        # kind of multimodal fusion (ave, concat, mul, sum)
        MULTIMODAL_MERGE_MODE = 'concat'

        model_config = Config(
            textual_embedding_dim=EMBEDDING_DIM,
            visual_embedding_dim=0,
            multimodal_merge_mode=MULTIMODAL_MERGE_MODE,
            input_dim=len(word2index_x.keys()),
            output_dim=len(word2index_y.keys()),
            visual_dim=train_visual_features.shape[1])
        model = VisionLanguageBOW(model_config)
        model.create()
```

```
        model.compile(
            loss='categorical_crossentropy',
            optimizer='adam')
```

In [ ]:
```
        #== Model training
        model.fit(
            train_input,
            train_y,
            batch_size=512,
            nb_epoch=40,
            validation_split=0.1,
            show_accuracy=True)
```

To achieve better results, we can use another operator that combines both modalities together. For instance, we can use a piece-wise multiplication.

In [ ]:
```
        #== Model definition

        # First we define a model using keras/kraino
        from keras.models import Sequential
        from keras.layers.core import Activation
        from keras.layers.core import Dense
        from keras.layers.core import Dropout
        from keras.layers.core import Layer
        from keras.layers.core import Merge
        from keras.layers.core import TimeDistributedMerge
        from keras.layers.embeddings import Embedding

        from kraino.core.model_zoo import AbstractSequentialModel
        from kraino.core.model_zoo import AbstractSingleAnswer
        from kraino.core.model_zoo import AbstractSequentialMultiplewordAnswer
        from kraino.core.model_zoo import Config
        from kraino.core.keras_extensions import DropMask
        from kraino.core.keras_extensions import LambdaWithMask
        from kraino.core.keras_extensions import time_distributed_masked_ave

        # This model inherits from AbstractSingleAnswer,
        # and so it produces single answer words
        # To use multiple answer words,
        # you need to inherit from AbstractSequentialMultiplewordAnswer
        class VisionLanguageBOW(AbstractSequentialModel, AbstractSingleAnswer):
            """
            BOW Language only model that produces single word answers.
            """
            def create(self):
```

```
                language_model = Sequential()
                language_model.add(Embedding(
                        self._config.input_dim,
                        self._config.textual_embedding_dim,
                        mask_zero=True))
                language_model.add(LambdaWithMask(
                        time_distributed_masked_ave,
                        output_shape=[language_model.output_shape[2]]))
                language_model.add(DropMask())
                visual_model = Sequential()
                if self._config.visual_embedding_dim > 0:
                    visual_model.add(Dense(
                            self._config.visual_embedding_dim,
                            input_shape=(self._config.visual_dim,)))
                else:
                    visual_model.add(Layer(input_shape=(self._config.visual_dim,)))
                self.add(Merge([language_model,
                            visual_model], mode=self._config.multimodal_merge_mode))
                self.add(Dropout(0.5))
                self.add(Dense(self._config.output_dim))
                self.add(Activation('softmax'))
```

In [ ]:
```
# dimensionality of embeddings
EMBEDDING_DIM = 500
# kind of multimodal fusion (ave, concat, mul, sum)
MULTIMODAL_MERGE_MODE = 'mul'

model_config = Config(
    textual_embedding_dim=EMBEDDING_DIM,
    visual_embedding_dim=EMBEDDING_DIM,
    multimodal_merge_mode=MULTIMODAL_MERGE_MODE,
    input_dim=len(word2index_x.keys()),
    output_dim=len(word2index_y.keys()),
    visual_dim=train_visual_features.shape[1])
model = VisionLanguageBOW(model_config)
model.create()
model.compile(
    loss='categorical_crossentropy',
    optimizer='adam')
text_image_bow_model = model
```

In [ ]:
```
#== Model training
text_image_bow_model.fit(
```

```
        train_input,
        train_y,
        batch_size=512,
        nb_epoch=40,
        validation_split=0.1,
        show_accuracy=True)
```

At the end of this section, a curious reader can try to answer the following questions.

```
If we merge language and visual features with 'mul',
do we need to set both embeddings to have the same number  of dimensions?
That is, do we require to have textual_embedding_dim == visual_embedding_dim?
```

C.8.0.2 RNN + Vision

Now, we repeat the BOW experiments but with RNN. Figure C.8 depicts the architecture.

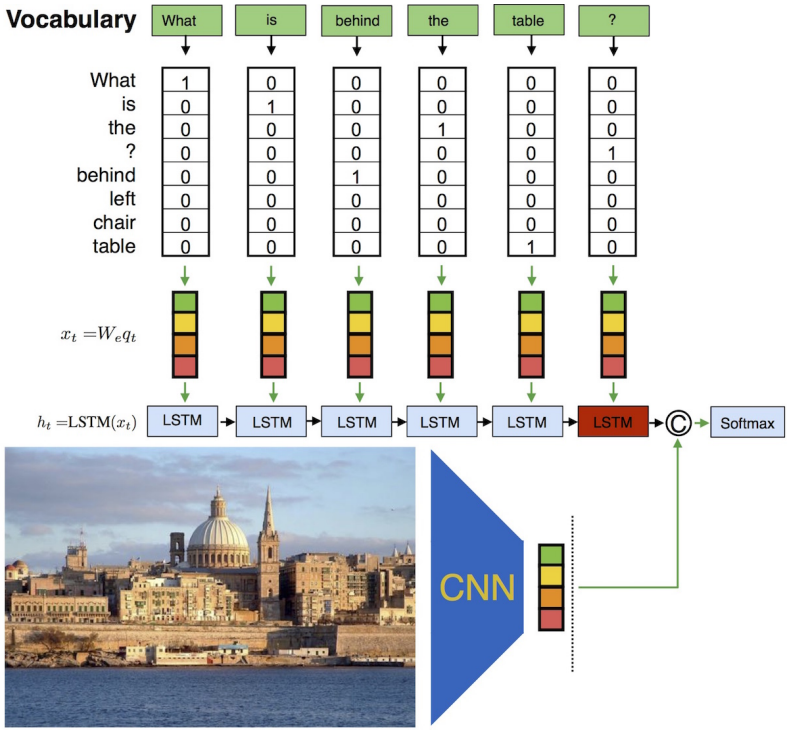


Figure C.8: RNN with visual features.

```
In [ ]: #== Model definition

        # First we define a model using keras/kraino
        from keras.models import Sequential
        from keras.layers.core import Activation
        from keras.layers.core import Dense
```

```
from keras.layers.core import Dropout
from keras.layers.core import Layer
from keras.layers.core import Merge
from keras.layers.core import TimeDistributedMerge
from keras.layers.embeddings import Embedding
from keras.layers.recurrent import GRU
from keras.layers.recurrent import LSTM

from kraino.core.model_zoo import AbstractSequentialModel
from kraino.core.model_zoo import AbstractSingleAnswer
from kraino.core.model_zoo import AbstractSequentialMultiplewordAnswer
from kraino.core.model_zoo import Config
from kraino.core.keras_extensions import DropMask
from kraino.core.keras_extensions import LambdaWithMask
from kraino.core.keras_extensions import time_distributed_masked_ave

# This model inherits from AbstractSingleAnswer,
# and so it produces single answer words
# To use multiple answer words,
# you need to inherit from AbstractSequentialMultiplewordAnswer
class VisionLanguageLSTM(AbstractSequentialModel, AbstractSingleAnswer):
    """
    BOW Language only model that produces single word answers.
    """
    def create(self):
        language_model = Sequential()
        language_model.add(Embedding(
                self._config.input_dim,
                self._config.textual_embedding_dim,
                mask_zero=True))

        language_model.add(LSTM(self._config.hidden_state_dim,
                      return_sequences=False))

        visual_model = Sequential()
        if self._config.visual_embedding_dim > 0:
            visual_model.add(Dense(
                    self._config.visual_embedding_dim,
                    input_shape=(self._config.visual_dim,)))
        else:
            visual_model.add(Layer(input_shape=(self._config.visual_dim,)))
        self.add(Merge([language_model,
                    visual_model], mode=self._config.multimodal_merge_mode))
```

```
            self.add(Dropout(0.5))
            self.add(Dense(self._config.output_dim))
            self.add(Activation('softmax'))


    # dimensionality of embeddings
    EMBEDDING_DIM = 500
    # kind of multimodal fusion (ave, concat, mul, sum)
    MULTIMODAL_MERGE_MODE = 'sum'

    model_config = Config(
        textual_embedding_dim=EMBEDDING_DIM,
        visual_embedding_dim=EMBEDDING_DIM,
        hidden_state_dim=EMBEDDING_DIM,
        multimodal_merge_mode=MULTIMODAL_MERGE_MODE,
        input_dim=len(word2index_x.keys()),
        output_dim=len(word2index_y.keys()),
        visual_dim=train_visual_features.shape[1])
    model = VisionLanguageLSTM(model_config)
    model.create()
    model.compile(
        loss='categorical_crossentropy',
        optimizer='adam')
    text_image_rnn_model = model
```

C.8.0.3 Batch Size

We can do training with batch size set to 512. If an error occurs due to a memory consumption, lowering the batch size should help.

In [ ]:
```
#== Model training
text_image_rnn_model.fit(
    train_input,
    train_y,
    batch_size=512,
    nb_epoch=40,
    validation_split=0.1,
    show_accuracy=True)
```

A curious reader may experiment with a few different batch sizes, and answer the following questions.

```
Can you experiment with batch-size=1, and next with batch-size=5000?
Can you explain both issues regarding the batch size?
When do you get the best performance, with multiplication, concatenation, or summation?
```

Summary As previously, using RNN makes the sequence processing order-aware. This time, however, we combine two modalities so that the whole model 'sees' images. Finally, it is also important how both modalities are combined.

C.9 New Predictions with Vision+Language

C.9.0.1 Predictions (Features)

```
In [ ]: test_image_names = test_text_representation['img_name']
        test_visual_features = dp['perception'](
            train_or_test='test',
            names_list=test_image_names,
            parts_extractor=None,
            max_parts=None,
            perception=CNN_NAME,
            layer=PERCEPTION_LAYER,
            second_layer=None
            )
        test_visual_features.shape
```

```
In [ ]: test_input = [test_x, test_visual_features]
```

C.9.0.2 Predictions (Bow with Vision)

```
In [ ]: from kraino.core.model_zoo import word_generator
        # we first need to add word_generator to _config
        # (we could have done this before, in the Config constructor)
        # we use maximum likelihood as a word generator
        text_image_bow_model._config.word_generator = word_generator['max_likelihood']
        predictions_answers = text_image_bow_model.decode_predictions(
            X=test_input,
            temperature=None,
            index2word=index2word_y,
            verbose=0)
```

```
In [ ]: _ = print_metrics.select['wups'](
                gt_list=test_raw_y,
                pred_list=predictions_answers,
                verbose=1,
                extra_vars=None)
```

C.9.0.3 Predictions (RNN with Vision)

```
In [ ]: from kraino.core.model_zoo import word_generator
        # we first need to add word_generator to _config
        # (we could have done this before, in the Config constructor)
```

```
        # we use maximum likelihood as a word generator
        text_image_rnn_model._config.word_generator = word_generator['max_likelihood']
        predictions_answers = text_image_rnn_model.decode_predictions(
            X=test_input,
            temperature=None,
            index2word=index2word_y,
            verbose=0)
```

```
In [ ]: _ = print_metrics.select['wups'](
                gt_list=test_raw_y,
                pred_list=predictions_answers,
                verbose=1,
                extra_vars=None)
```

C.10 VQA

The models that we have built so far can be transferred to other dataset. Let us consider a recently introduced large-scale dataset, which is named VQA [Antol et al. 2015]. In this section, we train and evaluate VQA models. Since the reader should already be familiar with all the pieces, we just quickly jump into coding. For the sake of simplicity, we use only BOW architectures. Since VQA hides the test data for the purpose of challenge, we use the publicly available validation set to evaluate the architectures.

C.10.0.1 VQA Language Features

```
In [ ]: #TODO: Execute the following procedure (Shift+Enter)
        from kraino.utils import data_provider

        vqa_dp = data_provider.select['vqa-real_images-open_ended']
        # VQA has a few answers associated with one question.
        # We take the most frequently occuring answers (single_frequent).
        # Formal argument 'keep_top_qa_pairs' allows to filter out
        # rare answers with the associated questions.
        # We use 0 as we want to keep all question answer pairs,
        # but you can change into 1000 and see how the results differ
        vqa_train_text_representation = vqa_dp['text'](
            train_or_test='train',
            answer_mode='single_frequent',
            keep_top_qa_pairs=1000)
        vqa_val_text_representation = vqa_dp['text'](
            train_or_test='val',
            answer_mode='single_frequent')
```

```
In [ ]: from toolz import frequencies
        vqa_train_raw_x = vqa_train_text_representation['x']
```

```
vqa_train_raw_y = vqa_train_text_representation['y']
vqa_val_raw_x = vqa_val_text_representation['x']
vqa_val_raw_y = vqa_val_text_representation['y']
# we start from building the frequencies table
vqa_wordcount_x = frequencies(' '.join(vqa_train_raw_x).split(' '))
# we can keep all answer words in the answer as a class
# therefore we use an artificial split symbol '{'
# to not split the answer into words
# you can see the difference if you replace '{'
# with ' ' and print vqa_wordcount_y
vqa_wordcount_y = frequencies('{'.join(vqa_train_raw_y).split('{'))
vqa_wordcount_y
```

C.10.0.2 Language-Only

```
In [ ]: from keras.preprocessing import sequence
        from kraino.utils.input_output_space import build_vocabulary
        from kraino.utils.input_output_space import encode_questions_index
        from kraino.utils.input_output_space import encode_answers_one_hot
        MAXLEN=30
        vqa_word2index_x, vqa_index2word_x = build_vocabulary
            (this_wordcount = vqa_wordcount_x)
        vqa_word2index_y, vqa_index2word_y = build_vocabulary
            (this_wordcount = vqa_wordcount_y)
        vqa_train_x = sequence.pad_sequences(encode_questions_index
            (vqa_train_raw_x, vqa_word2index_x), maxlen=MAXLEN)
        vqa_val_x = sequence.pad_sequences(encode_questions_index
            (vqa_val_raw_x, vqa_word2index_x), maxlen=MAXLEN)
        vqa_train_y, _ = encode_answers_one_hot(
            vqa_train_raw_y,
            vqa_word2index_y,
            answer_words_delimiter=vqa_train_text_representation['answer_words_delimiter']
            is_only_first_answer_word=True,
            max_answer_time_steps=1)
        vqa_val_y, _ = encode_answers_one_hot(
            vqa_val_raw_y,
            vqa_word2index_y,
            answer_words_delimiter=vqa_train_text_representation['answer_words_delimiter']
            is_only_first_answer_word=True,
            max_answer_time_steps=1)
```

```
In [ ]: from kraino.core.model_zoo import Config
        from kraino.core.model_zoo import word_generator
        # We are re-using the BlindBOW mode
```

```
        # Please make sure you have run the cell with the class definition
        # VQA is larger, so we can increase the dimensionality of the embedding
        vqa_model_config = Config(
            textual_embedding_dim=1000,
            input_dim=len(vqa_word2index_x.keys()),
            output_dim=len(vqa_word2index_y.keys()),
            word_generator = word_generator['max_likelihood'])
        vqa_text_bow_model = BlindBOW(vqa_model_config)
        vqa_text_bow_model.create()
        vqa_text_bow_model.compile(
            loss='categorical_crossentropy',
            optimizer='adam')
```

In [ ]:
```
vqa_text_bow_model.fit(
    vqa_train_x,
    vqa_train_y,
    batch_size=512,
    nb_epoch=10,
    validation_split=0.1,
    show_accuracy=True)

vqa_predictions_answers = vqa_text_bow_model.decode_predictions(
    X=vqa_val_x,
    temperature=None,
    index2word=vqa_index2word_y,
    verbose=0)

vqa_vars = {
    'question_id':vqa_val_text_representation['question_id'],
    'vqa_object':vqa_val_text_representation['vqa_object'],
    'resfun':
        lambda x: \
            vqa_val_text_representation['vqa_object'].loadRes(
                x, vqa_val_text_representation['questions_path'])
}
```

In [ ]:
```
from kraino.utils import print_metrics


_ = print_metrics.select['vqa'](
        gt_list=vqa_val_raw_y,
        pred_list=vqa_predictions_answers,
        verbose=1,
        extra_vars=vqa_vars)
```

C.10.0.3 VQA Language+Vision

In [ ]:
```
# the name for visual features that we use
VQA_CNN_NAME='vgg_net'
# VQA_CNN_NAME='googlenet'
# the layer in CNN that is used to extract features
VQA_PERCEPTION_LAYER='fc7'
# PERCEPTION_LAYER='pool5-7x7_s1'

vqa_train_visual_features = vqa_dp['perception'](
    train_or_test='train',
    names_list=vqa_train_text_representation['img_name'],
    parts_extractor=None,
    max_parts=None,
    perception=VQA_CNN_NAME,
    layer=VQA_PERCEPTION_LAYER,
    second_layer=None
    )
vqa_train_visual_features.shape
```

In [ ]:
```
vqa_val_visual_features = vqa_dp['perception'](
    train_or_test='val',
    names_list=vqa_val_text_representation['img_name'],
    parts_extractor=None,
    max_parts=None,
    perception=VQA_CNN_NAME,
    layer=VQA_PERCEPTION_LAYER,
    second_layer=None
    )
vqa_val_visual_features.shape
```

In [ ]:
```
from kraino.core.model_zoo import Config
from kraino.core.model_zoo import word_generator

# dimensionality of embeddings
VQA_EMBEDDING_DIM = 1000
# kind of multimodal fusion (ave, concat, mul, sum)
VQA_MULTIMODAL_MERGE_MODE = 'mul'

vqa_model_config = Config(
    textual_embedding_dim=VQA_EMBEDDING_DIM,
    visual_embedding_dim=VQA_EMBEDDING_DIM,
    multimodal_merge_mode=VQA_MULTIMODAL_MERGE_MODE,
    input_dim=len(vqa_word2index_x.keys()),
    output_dim=len(vqa_word2index_y.keys()),
```

```
            visual_dim=vqa_train_visual_features.shape[1],
            word_generator=word_generator['max_likelihood'])
        vqa_text_image_bow_model = VisionLanguageBOW(vqa_model_config)
        vqa_text_image_bow_model.create()
        vqa_text_image_bow_model.compile(
            loss='categorical_crossentropy',
            optimizer='adam')
```

In [ ]:
```
vqa_train_input = [vqa_train_x, vqa_train_visual_features]
vqa_val_input = [vqa_val_x, vqa_val_visual_features]
```

In [ ]:
```
#== Model training
vqa_text_image_bow_model.fit(
    vqa_train_input,
    vqa_train_y,
    batch_size=512,
    nb_epoch=10,
    validation_split=0.1,
    show_accuracy=True)

# we use maximum likelihood as a word generator
vqa_predictions_answers = vqa_text_image_bow_model.decode_predictions(
    X=vqa_val_input,
    temperature=None,
    index2word=vqa_index2word_y,
    verbose=0)

vqa_vars = {
    'question_id':vqa_val_text_representation['question_id'],
    'vqa_object':vqa_val_text_representation['vqa_object'],
    'resfun':
        lambda x: \
            vqa_val_text_representation['vqa_object'].loadRes(
                                x, vqa_val_text_representation['questions_path'])
}
```

In [ ]:
```
from kraino.utils import print_metrics


_ = print_metrics.select['vqa'](
        gt_list=vqa_val_raw_y,
        pred_list=vqa_predictions_answers,
        verbose=1,
        extra_vars=vqa_vars)
```

C.11 New Research Opportunities

The task that tests machines via questions about the content of images is a quite new research direction that recently has gained popularity. Therefore, many opportunities are available. We end the tutorial by enlisting a few possible directions.

- **Global Representation** In this tutorial, we use a global, full-frame representation of the images. Such a representation may destroy too much information. Therefore, it seems a fine-grained alternatives should be valid options. Maybe we should use detections, or object proposals (e.g. Ilievski et al. [2016] use question dependent detections, and Mokarian Forooshani et al. [2016] use object proposals to enrich a visual representation). We could also use attention models, which become quite successful in answering questions about images [Lu et al. 2016]. However, there is still a hope for global representations if they are trained end-to-end for the task, and question dependent. In the end, our global representation is extracted from CNNs trained on a different dataset (ImageNet), and for different task (object classification).
- **3D Scene Representation** Most of current approaches, and all neural-based approaches, are trained on 2D images. However, some spatial relations such as 'behind' may need a 3d representation of the scene (in fact Malinowski and Fritz [2014a] design spatial rules using a 3d coordinate system). DAQUAR is built on top of Silberman et al. [2012] that provides both modes (2D images, and 3D depth), however, such a richer visual information is currently not fully exploited.
- **Recurrent Neural Networks** There is disturbingly small gap between BOW and RNN models. As we have seen in the tutorial, some questions clearly require an order, but such questions at the same time become longer, semantically more difficult, and require better a visual understanding of the world. To handle them we may need other RNNs architectures, or better ways of fusing two modalities, or better **Global Representation**.
- **Logical Reasoning** There are few questions that require a bit more sophisticated logical reasoning such as negation. Can Recurrent Neural Networks learn such logical operators? What about compositionality of the language? Perhaps, we should aim at mixed approaches, similar to the work of Andreas et al. [2016b].
- **Language + Vision** There is a small gap between Language Only and Vision + Language models. But clearly, we need pictures to answer questions about images. So what is missing here? Is it due to **Global Representation**, **3D Scene Representation** or there is something missing in fusing two modalities? The latter is studied, with encouraging results, in Fukui et al. [2016].
- **Learning from Few Examples** In the Visual Turing Test, many questions are quite unique. But then how the models can generalize to new questions? What if a question is completely new, but its parts have been already observed (compositionality)? Can models guess the meaning of a new word from its context?
- **Ambiguities** How to deal with ambiguities? They are all inherent in the task, so cannot be just ignored, and should be incorporated into question answering methods as well as evaluation metrics.

- **Evaluation Measures** Although we have WUPS and Consensus, both are far from being perfect. Consensus has higher annotation cost for ambiguous tasks, and is unclear how to formally define good consensus measure. WUPS is an ontology dependent, which may be quite costly to build for all interesting domains? Finally, the current evaluation metrics ignore the tail of the answer distribution encouraging models to focus only on a few most frequent answers.

Bibliography

Gerald J Agin and Thomas O Binford. Computer description of curved objects. IEEE Transactions on Computers, 100(4):439–449, 1976. 16, 25

Zeynep Akata, Mateusz Malinowski, Mario Fritz, and Bernt Schiele. Multi-cue zero-shot learning with strong supervision. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016. 7, 22, 141

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Learning to compose neural networks for question answering. In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL), 2016a. 155

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural module networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016b. 4, 5, 51, 56, 59, 143, 167, 223

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. arXiv:1505.00468, 2015. 4, 55, 94, 120, 139, 140, 144, 145, 149, 150, 155, 164, 190, 218

Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 33(5):898–916, May 2011. ISSN 0162-8828. doi: 10.1109/TPAMI.2010.161. URL http://dx.doi.org/10.1109/TPAMI.2010.161. 94

Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. arXiv:1511.00561, 2015. 18

Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian J. Goodfellow, Arnaud Bergeron, Nicolas Bouchard, and Yoshua Bengio. Theano: new features and speed improvements. Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop, 2012. 150, 190, 196

Peter W Battaglia, Jessica B Hamrick, and Joshua B Tenenbaum. Simulation as an engine of physical scene understanding. Proceedings of the National Academy of Sciences (PNAS), 110(45):18327–18332, 2013. 94, 167

Islam Beltagy, Cuong Chau, Gemma Boleda, Dan Garrette, Katrin Erk, and Raymond Mooney. Montague meets markov: Deep semantics with probabilistic logical form. In *SEM, 2013. 97

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. IEEE transactions on neural networks, 5(2):157–166, 1994. 44

Jonathan Berant and Percy Liang. Semantic parsing via paraphrasing. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2014. 53, 94, 97, 99, 120, 142

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2013. 53, 98, 164

James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: a CPU and GPU math expression compiler. In Proceedings of the Python for Scientific Computing Conference (SciPy), 2010. 175

Apratim Bhattacharyya, Mateusz Malinowski, and Mario Fritz. Spatio-temporal image boundary extrapolation. arXiv:1605.07363, 2016a. 7, 167

Apratim Bhattacharyya, Mateusz Malinowski, Bernt Schiele, and Mario Fritz. Long-term image boundary extrapolation. arXiv:1611.08841, 2016b. 7, 167

Irving Biederman. Recognition-by-components: a theory of human image understanding. Psychological review, 94(2):115, 1987. 16, 25

C. M. Bishop. Neural Network for Pattern Recognition. Oxford University Press, 1999. 64

Patrick Blackburn and Johan Bos. Representation and Inference for Natural Language. A First Course in Computational Semantics. Center for the Study of Language and Information (CSLI), 2005. 93

Daniel G Bobrow. Natural language input for a computer problem solving system. PhD Thesis, 1964. 35, 52

Léon Bottou. Large-scale machine learning with stochastic gradient descent. In COMPSTAT'2010. 2010. 39

Léon Bottou. Stochastic gradient descent tricks. In Neural Networks: Tricks of the Trade, pages 421–436. Springer, 2012. 174

Çaglar Gülçehre and Yoshua Bengio. Knowledge matters: Importance of prior information for optimization. Computing Research Repository (CoRR), 2013. 175

Kan Chen, Jiang Wang, Liang-Chieh Chen, Haoyuan Gao, Wei Xu, and Ram Nevatia. Abc-cnn: An attention based convolutional neural network for visual question answering. arXiv:1511.05960, 2015. 4, 58, 143, 156

Yixin Chen, Jinbo Bi, and James Ze Wang. Miles: Multiple-instance learning via embedded instance selection. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 28(12):1931–1947, 2006. 79

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Fethi Bougares, Holger Schwenk, Dzmitry Bahdanau, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014. 43, 44, 52, 93, 119, 141, 146, 152, 200

Eunsol Choi, Tom Kwiatkowski, and Luke S Zettlemoyer. Scalable semantic parsing with partial ontologies. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2015. 164

Myung Jin Choi, Joseph J. Lim, Antonio Torralba, and Alan S. Willsky. Exploiting hierarchical context on a large database of object categories. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010. 81, 175

François Chollet. keras. https://github.com/fchollet/keras, 2015. 150, 190, 197

Sreyasi Nag Chowdhury. Commonsense for making sense of data. 2016. 165

Sreyasi Nag Chowdhury, Mateusz Malinowski, Andreas Bulling, and Mario Fritz. Xplore-m-ego: Contextual media retrieval using natural language queries. In ACM International Conference in Multimedia Retrieval (ICMR), 2016a. 4, 6, 8, 51, 55, 145, 163, 165, 167

Sreyasi Nag Chowdhury, Niket Tandon, and Gerhard Weikum. Know2look: Commonsense knowledge for visual search. 2016b. 165

Gordon Christie, Ankit Laddha, Aishwarya Agrawal, Stanislaw Antol, Yash Goyal, Kevin Kochersberger, and Dhruv Batra. Resolving language and vision ambiguities together: Joint segmentation & prepositional attachment resolution in captioned scenes. arXiv:1604.02125, 2016. 50

James Clarke, Dan Goldwasser, Ming-Wei Chang, and Dan Roth. Driving semantic parsing from the world's response. In Proceedings of the Fifteenth Conference on Computational Natural Language Learning (CoNLL), 2010. 37

A. Coates and A. Y. Ng. The importance of encoding versus training with sparse coding and vector quantization. In Proceedings of the International Conference on Machine Learning (ICML), 2011. 26, 62, 67, 69, 70, 71, 251

A. Coates, H. Lee, and A. Y. Ng. An analysis of single-layer networks in unsupervised feature learning. In International Conference on Artificial Intelligence and Statistics (AISTATS), 2011. 67

Jacob Cohen et al. A coefficient of agreement for nominal scales. Educational and psychological measurement, 1960. 126

R. Collobert and S. Bengio. Links between perceptrons, mlps and svms. In Proceedings of the International Conference on Machine Learning (ICML), 2004. 64

Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In Workshop on statistical learning in computer vision, ECCV, 2004. 16, 18

N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2005. 16, 27

Marie-Catherine De Marneffe, Bill MacCartney, and Christopher D Manning. Generating typed dependency parses from phrase structure parses. In International Conference on Language Resources and Evaluation (LREC), 2006. 79

Jia Deng, Alexander C Berg, Kai Li, and Li Fei-Fei. What does classifying more than 10,000 image categories tell us? In Proceedings of the European Conference on Computer Vision (ECCV), 2010. 96

Jeff Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015. 22, 53, 119, 122, 123, 141, 149

Jeffrey L Elman. Finding structure in time. Cognitive science, 14(2):179–211, 1990. 43

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? Journal of Machine Learning Research (JMLR), 11, 2010. 175

M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. International Journal of Computer Vision (IJCV), 111(1):98–136, jan 2015. 17

Mark Everingham, Luc Van Gool, CKI Williams, John Winn, and Andrew Zisserman. Pascal 2008 results, 2008. 81

Abdalrahman Eweiwi, Muhammad Shahzad Cheema, and Christian Bauckhage. Action recognition in still images by learning spatial interest regions from videos. Pattern Recognition Letters, 51:8–15, 2015. 48

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. Open question answering over curated and extracted knowledge bases. In Proceedings of the ACM SIGKDD international conference on Knowledge discovery and data mining, 2014. 94

Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth. Every picture tells a story: Generating sentences from images. In Proceedings of the European Conference on Computer Vision (ECCV). 2010. 94, 172

Christiane Fellbaum. WordNet. Wiley Online Library, 1999. 99, 110, 203

P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2008. 67, 71

P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2010. 83, 84, 169, 173, 178

J. Feng, B. Ni, Q. Tian, and S. Yan. Geometric lp-norm feature pooling for image classification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2011. 48, 62

Joseph L Fleiss and Jacob Cohen. The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. Educational and psychological measurement, 1973. 126

Jerry A Fodor. The language of thought. Harvard University Press, 1975. 19

Katerina Fragkiadaki, Pulkit Agrawal, Sergey Levine, and Jitendra Malik. Learning visual predictive models of physics for playing billiards. International Conference on Learning Representations (ICLR), 2015. 167

Mario Fritz, Geert-Jan M Kruijff, and Bernt Schiele. Cross-modal learning of visual categories using different levels of supervision. In Internation Conference on Vision Systems (ICVS), 2007. 49, 53

Mario Fritz, Geert-Jan M Kruijff, and Bernt Schiele. Tutor-based learning of visual categories using different levels of supervision. Computer Vision and Image Understanding, 114(5): 564–573, 2010. 49, 53

Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Tomas Mikolov, et al. Devise: A deep visual-semantic embedding model. In Advances in Neural Information Processing Systems (NIPS), 2013. 86

Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. Multimodal compact bilinear pooling for visual question answering and visual grounding. arXiv:1606.01847, 2016. 58, 165, 166, 167, 223

K. Fukushima and S. Miyake. Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position. Pattern recognition, 15(6):455–469, 1982. 62

Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological cybernetics, 36(4):193–202, 1980. 28

Haoyuan Gao, Junhua Mao, Jie Zhou, Zhiheng Huang, Lei Wang, and Wei Xu. Are you talking to a machine? dataset and methods for multilingual image question answering. In Advances in Neural Information Processing Systems (NIPS), 2015. 4, 55, 58, 120, 142, 144

Donald Geman, Stuart Geman, Neil Hallonquist, and Laurent Younes. Visual turing test for computer vision systems. Proceedings of the National Academy of Sciences (PNAS), 2015. 4, 55, 120, 144

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 580–587, 2014. 1, 17, 79, 84, 93, 100, 161

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier networks. In International Conference on Artificial Intelligence and Statistics (AISTATS), 2011. 79

Dave Golland, Percy Liang, and Dan Klein. A game-theoretic approach to generating spatial descriptions. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2010. 50, 76, 85, 97, 99, 171

Yunchao Gong, Liwei Wang, Micah Hodosh, Julia Hockenmaier, and Svetlana Lazebnik. Improving image-sentence embeddings using large weakly annotated photo collections. In Proceedings of the European Conference on Computer Vision (ECCV). 2014. 98

I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio. Maxout networks. In Proceedings of the International Conference on Machine Learning (ICML), 2013. 49, 67, 70

S. Guadarrama, N. Krishnamoorthy, G. Malkarnenkar, R. Mooney, T. Darrell, and K. Saenko. Youtube2text: Recognizing and describing arbitrary activities using semantic hierarchies and zero-shot recognition. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2013a. 110

Sergio Guadarrama, Lorenzo Riano, Dave Golland, Daniel Gouhring, Yangqing Jia, Dan Klein, Pieter Abbeel, and Trevor Darrell. Grounding spatial relations for human-robot interaction. In International Conference on Intelligent Robots and Systems (IROS), 2013b. 50, 74, 76, 108

Saurabh Gupta, Pablo Arbelaez, and Jitendra Malik. Perceptual organization and recognition of indoor scenes from rgb-d images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013. 98, 107, 109, 111, 170

Saurabh Gupta, Ross Girshick, Pablo Arbeláez, and Jitendra Malik. Learning rich features from rgb-d images for object detection and segmentation. In Proceedings of the European Conference on Computer Vision (ECCV). 2014. 93, 98

Stevan Harnad. The symbol grounding problem. Physica D: Nonlinear Phenomena, 42(1): 335–346, 1990. 98

James Hays and Alexei A Efros. Scene completion using millions of photographs. In ACM Transactions on Graphics (TOG), volume 26, page 4. ACM, 2007. 15

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In Proceedings of the European Conference on Computer Vision (ECCV). 2014. 93

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. arXiv:1512.03385, 2015. 1, 31, 34, 52, 140, 141, 148, 154, 161, 165

Yang He, Wei-Chen Chiu, Margret Keuper, and Mario Fritz. Rgbd semantic segmentation using spatio-temporal data-driven pooling. arXiv:1604.02388, 2016. 49

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. arXiv:1207.0580, 2012. 30

Sepp Hochreiter. Untersuchungen zu dynamischen neuronalen netzen. Diploma, Technische Universität München, page 91, 1991. 44

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. Neural Computation, 1997. 43, 44, 52, 119, 122, 140, 141, 146, 152, 184, 200

Micah Hodosh, Peter Young, and Julia Hockenmaier. Framing image description as a ranking task: Data, models and evaluation metrics. Journal of Artificial Intelligence Research (JAIR), 47:853–899, 2013. 74, 81, 94

Judy Hoffman, Sergio Guadarrama, Eric Tzeng, Ronghang Hu, Jeff Donahue, Ross Girshick, Trevor Darrell, and Kate Saenko. LSDA: Large scale detection through adaptation. In Advances in Neural Information Processing Systems (NIPS), 2014. 96

John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. volume 79, pages 2554–2558. Proceedings of the National Academy of Sciences (PNAS), 1982. 43

Ronghang Hu, Huazhe Xu, Marcus Rohrbach, Jiashi Feng, Kate Saenko, and Trevor Darrell. Natural language object retrieval. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016. 141

Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. Improving word representations via global context and multiple word prototypes. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2012. 79

D. H. Hubel and T. N. Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. The Journal of physiology, 160(1):106, 1962. 28, 62

Ilija Ilievski, Shuicheng Yan, and Jiashi Feng. A focused dynamic attention model for visual question answering. arXiv:1604.01485, 2016. 4, 58, 143, 155, 166, 223

Mohit Iyyer, Jordan Boyd-Graber, Leonardo Claudino, Richard Socher, and Hal Daumé III. A neural network for factoid question answering over paragraphs. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014. 53, 94, 97, 120, 142

Y. Jia and C. Huang. Beyond spatial pyramids: Receptive field learning for pooled image features. In NIPS Workshop on Deep Learning, 2011. 48, 62, 67, 69, 70, 251

Y. Jia, C. Huang, and T. Darrell. Beyond spatial pyramids: Receptive field learning for pooled image features. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012. 48, 62, 68

Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. arXiv:1408.5093, 2014. 123, 148, 208

Aiwen Jiang, Fang Wang, Fatih Porikli, and Yi Li. Compositional memory for visual question answering. arXiv:1511.05676, 2015. 155, 156

Michael I Jordan. Serial order: A parallel distrmuted processing approach. 1986. 43

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2014. 52, 140, 141, 147

Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015. 53, 118, 119, 120, 141

Andrej Karpathy, Armand Joulin, and Li Fei-Fei. Deep fragment embeddings for bidirectional image sentence mapping. In Advances in Neural Information Processing Systems (NIPS), 2014. 7, 11, 22, 50, 73, 75, 76, 77, 79, 80, 81, 82, 84, 86, 87, 94, 105, 120, 141

John D Kelleher, Geert-Jan M Kruijff, and Fintan J Costello. Proximity in context: an empirically grounded computational model of proximity for processing topological spatial expressions. In COLING-ACL, 2006. 171

Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. A diagram is worth a dozen images. Proceedings of the European Conference on Computer Vision (ECCV), 2016. 164

Yoon Kim. Convolutional neural networks for sentence classification. Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014. 52, 140, 141, 147

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv:1412.6980, 2014. 150, 196

Dan Klein and Christopher D Manning. Accurate unlexicalized parsing. In Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1, pages 423–430. Association for Computational Linguistics, 2003. 55, 144, 173

Jan J Koenderink and Andrea J Van Doorn. The structure of locally orderless images. International Journal of Computer Vision (IJCV), 31(2-3):159–168, 1999. 26, 62

Chen Kong, Dahua Lin, Mohit Bansal, Raquel Urtasun, and Sanja Fidler. What are you talking about? text-to-image coreference. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014. 50, 76, 94, 98, 105, 120, 141

P. Koniusz and K. Mikolajczyk. Spatial coordinate coding to reduce histogram representations, dominant angle and colour pyramid match. In International Conference on Image Processing, 2011. 48, 62, 63

J. Krapac, J. Verbeek, and F. Jurie. Modeling spatial layout with fisher vectors for image categorization. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2011. 48, 62, 63

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, Michael Bernstein, and Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. arXiv:1602.07332, 2016. 144

Jayant Krishnamurthy and Thomas Kollar. Jointly learning to parse and perceive: Connecting natural language to the physical world. Transactions of the Association for Computational Linguistics (TACL), 2013. 53, 54, 94, 98, 105, 107, 120, 141

A. Krizhevsky and G. Hinton. Convolutional deep belief networks on cifar-10. Technical report, 2010. 66

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In Advances in Neural Information Processing Systems (NIPS), 2012. 1, 16, 29, 30, 31, 50, 51, 52, 76, 93, 118, 119, 123, 140, 141, 148, 154, 161, 164, 169, 172

Geert-Jan M Kruijff, Hendrik Zender, Patric Jensfelt, and Henrik I Christensen. Situated dialogue and spatial organization: What, where... and why. International Journal of Advanced Robotic Systems (IJARS), 2007. 50, 76, 105

Girish Kulkarni, Visruth Premraj, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C Berg, and Tamara L Berg. Baby talk: Understanding and generating simple image descriptions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2011. 94, 172

Ankit Kumar, Ozan Irsoy, Jonathan Su, James Bradbury, Robert English, Brian Pierce, Peter Ondruska, Ishaan Gulrajani, and Richard Socher. Ask me anything: Dynamic memory networks for natural language processing. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016. 58, 143

Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. Inducing probabilistic ccg grammars from logical form with higher-order unification. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2010. 37, 52, 93, 97, 104

Katrina LaCurts. Criticisms of the turing test and why you should ignore (most of) them. 2011. 94

Brenden M Lake, Ruslan Salakhutdinov, and Josh Tenenbaum. One-shot learning by inverting a compositional causal process. In Advances in Neural Information Processing Systems (NIPS), 2013. 94

George Lakoff. Women, fire, and dangerous things: What categories reveal about the mind. Cambridge University Press, 1990. 96, 97

Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 951–958. IEEE, 2009. 22

Tian Lan, Weilong Yang, Yang Wang, and Greg Mori. Image retrieval with structured object queries using latent ranking svm. In Proceedings of the European Conference on Computer Vision (ECCV). 2012. 22, 49, 50, 74, 75, 76, 77, 80, 81, 82, 83, 84, 98, 108, 162, 170, 171, 175, 176, 177, 178, 251, 253

Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2006. 16, 18, 26, 27, 47, 48, 50, 62, 63, 67, 75, 76, 77

Q. V. Le, M. A. Ranzato, R. Monga, M. Devin, K. Chen, G. S. Corrado, J. Dean, and A. Y. Ng. Building high-level features using large scale unsupervised learning. In Proceedings of the International Conference on Machine Learning (ICML), 2012. 62, 66

Vadim Lebedev and Victor Lempitsky. Fast convnets using group-wise brain damage. arXiv:1506.02515, 2015. 49

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Handwritten digit recognition with a back-propagation network. In Advances in Neural Information Processing Systems (NIPS), 1990. 62

Y. LeCun, L. Bottou, G. Orr, and K. Müller. Efficient backprop. Neural networks: Tricks of the trade, pages 546–546, 1998a. 28, 29, 64, 174

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 1998b. 28, 119, 122

Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. arXiv:1409.5185, 2014. 93

Chen-Yu Lee, Patrick W Gallagher, and Zhuowen Tu. Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree. In International Conference on Artificial Intelligence and Statistics (AISTATS), 2016. 5, 48, 167

Hector J Levesque. The winograd schema challenge. 2011. 19

Stephen C Levinson. Space in language and cognition: Explorations in cognitive diversity, volume 5. Cambridge University Press, 2003. 97

Michael Levit and Deb Roy. Interpretation of spatial language in a map navigation task. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, 2007. 105

Mike Lewis and Mark Steedman. Combining formal and distributional models of temporal and intensional semantics. In ACL Workshop on Semantic Parsing, 2014. 97

Chi Li, Austin Reiter, and Gregory D Hager. Beyond spatial pooling: fine-grained representation learning in multiple domains. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 4913–4922, 2015. 49

Li-Jia Li, Hao Su, Eric P Xing, and Li Fei-Fei. Object bank: A high-level image representation for scene classification and semantic feature sparsification. Advances in Neural Information Processing Systems (NIPS), 2010. 169, 173

L. Li-Jia and L. Fei-Fei. What, where and who? classifying events by scene and object recognition. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2007. 66

L. Li-Jia, S. Hao, E. P. Xing, and L. Fei-Fei. Object bank: A high-level image representation for scene classification and semantic feature sparsification. In Advances in Neural Information Processing Systems (NIPS), 2010. 62, 63, 66, 67, 71

Percy Liang. Talking to computers in natural language. XRDS: Crossroads, The ACM Magazine for Students, 2014. 37

Percy Liang and Christopher Potts. Bringing machine learning and compositional semantics together. Annual Review of Linguistics, 2015. 36, 37, 38, 39, 251

Percy Liang, Michael I Jordan, and Dan Klein. Learning dependency-based compositional semantics. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2011. 171

Percy Liang, Michael I Jordan, and Dan Klein. Learning dependency-based compositional semantics. Computational Linguistics, 2013. 5, 20, 36, 37, 40, 41, 42, 52, 53, 94, 97, 99, 104, 106, 107, 120, 142, 145, 166

Dahua Lin, Sanja Fidler, and Raquel Urtasun. Holistic scene understanding for 3d object detection with rgbd cameras. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2013. 98, 170

Dahua Lin, Sanja Fidler, Chen Kong, and Raquel Urtasun. Visual semantic search: Retrieving videos via complex textual queries. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014a. 50, 76

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

Proceedings of the European Conference on Computer Vision (ECCV), 2014b. 55, 144, 150

Yinglu Liu, Yan-Ming Zhang, Xu-Yao Zhang, and Cheng-Lin Liu. Adaptive spatial pooling for image classification. Pattern Recognition, 55:58–67, 2016. 48

Gordon D Logan and Daniel D Sadler. A computational analysis of the apprehension of spatial relations. 1996. 6, 49, 75, 76, 77, 79, 171, 172, 174

D. G. Lowe. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision (IJCV), 60(2):91–110, 2004. 16, 27

Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. Hierarchical question-image co-attention for visual question answering. 2016. 58, 166, 223

Lin Ma, Zhengdong Lu, and Hang Li. Learning to answer questions from image using convolutional neural network. arXiv:1506.00333, 2015. 58, 120, 142, 147, 148, 156

Mateusz Malinowski and Mario Fritz. Learning smooth pooling regions for visual recognition. In Proceedings of the British Machine Vision Conference (BMVC), 2013a. 5, 6, 9, 14, 27, 48, 49, 75, 76, 77

Mateusz Malinowski and Mario Fritz. Learnable pooling regions for image classification. In ICLR Workshop, 2013b. 5, 6, 9, 14, 27, 48, 49, 62

Mateusz Malinowski and Mario Fritz. A multi-world approach to question answering about real-world scenes based on uncertain input. In Advances in Neural Information Processing Systems (NIPS), 2014a. 3, 4, 5, 6, 10, 11, 14, 40, 50, 51, 54, 55, 57, 59, 74, 94, 95, 96, 97, 98, 99, 100, 101, 119, 120, 123, 124, 125, 127, 130, 131, 133, 140, 142, 143, 144, 145, 163, 164, 165, 166, 167, 183, 187, 190, 191, 203, 204, 223

Mateusz Malinowski and Mario Fritz. Towards a visual turing challenge. In NIPS workshop on Learning Semantics, 2014b. 3, 5, 6, 10, 14, 51, 55, 57, 120, 140, 142, 144, 163, 165, 183, 203

Mateusz Malinowski and Mario Fritz. A pooling approach to modelling spatial relations for image retrieval and annotation. arXiv:1411.5190, 2014c. 4, 5, 6, 10, 14, 22, 49, 59, 143, 165

Mateusz Malinowski and Mario Fritz. Hard to cheat: A turing test based on answering questions about images. AAAI Workshop: Beyond the Turing Test, 2015. 3, 5, 10, 14, 51, 55, 95, 140, 144, 163, 183, 203

Mateusz Malinowski and Mario Fritz. Tutorial on answering questions about images with deep learning. arXiv:1610.01076, 2016. 11, 14

Mateusz Malinowski, Marcus Rohrbach, and Mario Fritz. Ask your neurons: A neural-based approach to answering questions about images. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015. 4, 5, 6, 11, 14, 51, 55, 58, 94, 140, 142, 145, 149, 156, 165, 166, 186, 187, 190, 196, 203, 205, 253

Mateusz Malinowski, Marcus Rohrbach, and Mario Fritz. Ask your neurons: A deep learning approach to visual question answering. arXiv:1605.02697, 2016. 4, 5, 6, 14, 18, 25, 58, 155, 164, 166, 190, 253

Christopher D Manning and Hinrich Schütze. Foundations of statistical natural language processing, volume 999. MIT Press, 1999. 141, 171, 173

Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. Introduction to information retrieval. Cambridge university press Cambridge, 2008. 108, 140

Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, and Alan L Yuille. Explain images with multimodal recurrent neural networks. arXiv:1410.1090, 2014. 50, 76, 94

Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016. 141

Cynthia Matuszek, Nicholas Fitzgerald, Luke Zettlemoyer, Liefeng Bo, and Dieter Fox. A joint model of language and perception for grounded attribute learning. In Proceedings of the International Conference on Machine Learning (ICML), 2012. 53, 94, 98, 105, 120, 141, 172

Cynthia Matuszek, Evan Herbst, Luke Zettlemoyer, and Dieter Fox. Learning to parse natural language commands to a robot control system. In Experimental Robotics, 2013. 105

Michael McCloskey. Intuitive physics. Scientific American, 1984. 167

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In Advances in Neural Information Processing Systems (NIPS), 2013. 20, 93, 97, 100

George A Miller. Wordnet: a lexical database for english. CACM, 1995. 99, 110, 203

Ashkan Mokarian, Mateusz Malinowski, and Mario Fritz. Mean box pooling: A rich image representation and output embedding for the visual madlibs task. Proceedings of the British Machine Vision Conference (BMVC), 2016. 51

Ashkan Mokarian Forooshani, Mateusz Malinowski, and Mario Fritz. Mean box pooling: A rich image representation and output embedding for the visual madlibs task. In Proceedings of the British Machine Vision Conference (BMVC). BMVA Press, 2016. 8, 58, 166, 223

Reinhard Moratz and Thora Tenbrink. Spatial reference in linguistic human-robot interaction: Iterative, empirically supported development of a model of projective relations. Spatial cognition and computation, 2006. 50, 76, 171

Roozbeh Mottaghi, Mohammad Rastegari, Abhinav Gupta, and Ali Farhadi. " what happens if..." learning to predict the effect of forces in images. Proceedings of the European Conference on Computer Vision (ECCV), 2016. 167

Joseph L Mundy. Object recognition in the geometric era: A retrospective. In Toward category-level object recognition, pages 3–28. Springer, 2006. 16

Hiroshi Murase and Shree K Nayar. Visual learning and recognition of 3-d objects from appearance. International Journal of Computer Vision (IJCV), 14(1):5–24, 1995. 16, 25

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In Proceedings of the International Conference on Machine Learning (ICML), 2010. 29

Ndapandula Nakashole, Tomasz Tylenda, and Gerhard Weikum. Fine-grained semantic typing of emerging entities. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2013. 126

Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), pages 1520–1528, 2015a. 18

Hyeonwoo Noh, Paul Hongsuck Seo, and Bohyung Han. Image question answering using convolutional neural network with dynamic parameter prediction. arXiv:1511.05756, 2015b. 59, 143, 155

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014. 100, 148, 152

Florent Perronnin, Zeynep Akata, Zaid Harchaoui, and Cordelia Schmid. Towards good practice in large-scale learning for image classification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012. 96

Leonid Pishchulin, Mykhaylo Andriluka, Peter Gehler, and Bernt Schiele. Strong appearance and expressive spatial models for human pose estimation. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2013. 93

Tomaso Poggio and Shimon Edelman. A network that learns to recognize 3d objects. Nature, 343(6255):263–266, 1990. 25

M. A. Ranzato, F. J. Huang, Y. Boureau, and Y. LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2007. 62

Cyrus Rashtchian, Peter Young, Micah Hodosh, and Julia Hockenmaier. Collecting image annotations using amazon's mechanical turk. In NAACL HLT Workshop, 2010. 81, 82, 98

Terry Regier and Laura A Carlson. Grounding spatial language in perception: an empirical and computational investigation. Journal of Experimental Psychology: General, 2001. 75, 108

Michaela Regneri, Marcus Rohrbach, Dominikus Wetzel, Stefan Thater, Bernt Schiele, and Manfred Pinkal. Grounding Action Descriptions in Videos. Transactions of the Association for Computational Linguistics (TACL), 1, 2013. 56, 145

Mengye Ren, Ryan Kiros, and Richard Zemel. Image question answering: A visual semantic embedding model and a new dataset. In Advances in Neural Information Processing Systems (NIPS), 2015a. 4, 55, 58, 120, 142, 144

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In Advances in Neural Information Processing Systems (NIPS), 2015b. 1, 18, 25, 161

M. Riesenhuber and T. Poggio. Hierarchical models of object recognition in cortex. Nature Neuroscience, 2009. 62

Lawrence Gilman Roberts. Machine perception of three-dimensional soups. PhD thesis, Massachusetts Institute of Technology, 1963. 16, 25

Tim Rocktäschel, Matko Bosnjak, Sameer Singh, and Sebastian Riedel. Low-dimensional embeddings of logic. In ACL Workshop on Semantic Parsing, 2014. 97

Anna Rohrbach, Marcus Rohrbach, Wei Qiu, Annemarie Friedrich, Sikandar Amin, Mykhaylo Andriluka, Manfred Pinkal, and Bernt Schiele. Coherent multi-sentence video description with variable level of detail. In Proceedings of the German Confeence on Pattern Recognition (GCPR), 2014. 94

Anna Rohrbach, Marcus Rohrbach, Ronghang Hu, Trevor Darrell, and Bernt Schiele. Grounding of textual phrases in images by reconstruction. arXiv:1511.03745, 2015a. 141

Anna Rohrbach, Marcus Rohrbach, Niket Tandon, and Bernt Schiele. A dataset for movie description. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015b. 56, 145

Marcus Rohrbach, Michael Stark, and Bernt Schiele. Evaluating knowledge transfer and zero-shot learning in a large-scale setting. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2011. 22, 98

Marcus Rohrbach, Michaela Regneri, Mykhaylo Andriluka, Sikandar Amin, Manfred Pinkal, and Bernt Schiele. Script data for attribute-based recognition of composite activities. In Proceedings of the European Conference on Computer Vision (ECCV). 2012. 98

Eleanor H Rosch. Natural categories. Cognitive psychology, 4(3):328–350, 1973. 97

Olga Russakovsky, Yuanqing Lin, Kai Yu, and Li Fei-Fei. Object-centric spatial pooling for image classification. In Proceedings of the European Conference on Computer Vision (ECCV). 2012. 48, 50, 62, 63, 75, 76

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. arXiv:1409.0575, 2014. 14, 15, 22, 29, 52, 98, 119, 123, 140, 148, 165, 208

Fereshteh Sadeghi, Santosh K Divvala, and Ali Farhadi. Viske: Visual knowledge extraction and question answering by visual verification of relation phrases. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1456–1464, 2015. 165

J. Sánchez, F. Perronnin, and T. de Campos. Modeling the spatial layout of images beyond spatial pyramids. Pattern Recognition Letters, 2012. 48, 62, 63

Bernhard Schölkopf and Christopher JC Burges. Advances in kernel methods: support vector learning. MIT press, 1999. 39

Thomas Serre, Lior Wolf, Stanley Bileschi, Maximilian Riesenhuber, and Tomaso Poggio. Robust object recognition with cortex-like mechanisms. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 29(3):411–426, 2007. 28

Qi Shan, Riley Adams, Brian Curless, Yasutaka Furukawa, and Steven M Seitz. The visual turing test for scene reconstruction. In International Conference on 3D Vision (3DV), 2013. 94

Gaurav Sharma and Frederic Jurie. Learning discriminative spatial representation for image classification. In Proceedings of the British Machine Vision Conference (BMVC), pages 1–11. BMVA Press, 2011. 48

Kevin J Shih, Saurabh Singh, and Derek Hoiem. Where to look: Focus regions for visual question answering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016. 4, 58, 143

Behjat Siddiquie, Rogério Schmidt Feris, and Larry S Davis. Image ranking and retrieval based on multi-attribute queries. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2011. 83, 171, 178

Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In Proceedings of the European Conference on Computer Vision (ECCV), 2012. 53, 98, 105, 109, 123, 144, 223

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. International Conference on Learning Representations (ICLR), 2015. 31, 52, 93, 119, 141, 148, 154, 208

Gudrun Socher, Gerhard Sagerer, and Pietro Perona. Bayesian reasoning on qualitative descriptions from images and speech. Image Vision Computing, 2000. 172

Richard Socher, Cliff C Lin, Andrew Y Ng, and Christopher D Manning. Parsing natural scenes and natural language with recursive neural networks. In Proceedings of the International Conference on Machine Learning (ICML), 2011. 5, 170, 172

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2013. 51

Richard Socher, Andrej Karpathy, Q Le, C Manning, and A Ng. Grounded compositional semantics for finding and describing images with sentences. In Transactions of the Association for Computational Linguistics (TACL), 2014. 50, 76, 81, 86, 94, 97

Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. In Advances in Neural Information Processing Systems (NIPS), pages 2440–2448, 2015. 183, 184, 187

Ilya Sutskever, Oriol Vinyals, and Quoc V. V Le. Sequence to sequence learning with neural networks. In Advances in Neural Information Processing Systems (NIPS). 2014. 52, 118, 119, 122, 141

Michael J Swain and Dana H Ballard. Color indexing. International Journal of Computer Vision (IJCV), 7(1):11–32, 1991. 16, 25

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015. 1, 31, 32, 33, 34, 52, 119, 123, 141, 148, 154, 161

Niket Tandon. Commonsense knowledge acquisition and applications. 2016. 165

Makarand Tapaswi, Yukun Zhu, Rainer Stiefelhagen, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Movieqa: Understanding stories in movies through question-answering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016. 4, 56, 145

Stefanie Tellex, Thomas Kollar, George Shaw, Nicholas Roy, and Deb Roy. Grounding spatial language for video search. In International Conference on Multimodal Interaction (ICMI), 2010. 50, 76, 171

Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew R Walter, Ashis Gopal Banerjee, Seth J Teller, and Nicholas Roy. Understanding natural language commands for robotic navigation and mobile manipulation. In Proceedings of the Conference on Artificial Intelligence (AAAI), 2011. 36, 53, 74, 105

Tatiana Tommasi, Arun Mallya, Bryan Plummer, Svetlana Lazebnik, Alexander C Berg, and Tamara L Berg. Solving visual madlibs with multiple cues. In Proceedings of the British Machine Vision Conference (BMVC). BMVA Press, 2016. 58, 166

Jonathan Tompson, Arjun Jain, Yann LeCun, and Christoph Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In Advances in Neural Information Processing Systems (NIPS), 2014. 93

A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2008. 66

Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In HLT-NAACL, 2003. 98

Trecvid. Trecvid med 14. http://nist.gov/itl/iad/mig/med14.cfm, 2014. 56, 145

John W Tukey. Exploratory data analysis. 1977. 110

Alan M Turing. Computing machinery and intelligence. Mind, pages 433–460, 1950. 14, 94, 95, 163

Joost Van De Weijer, Cordelia Schmid, and Jakob Verbeek. Learning color names from real-world images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2007. 107, 109, 112, 186

J. C. van Gemert. Exploiting photographic style for category-level image classification by generalizing the spatial pyramid. In ACM International Conference in Multimedia Retrieval (ICMR), 2011. 63

Ramakrishna Vedantam, Xiao Lin, Tanmay Batra, C. Lawrence Zitnick, and Devi Parikh. Learning common sense through visual abstraction. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015. 165

Subhashini Venugopalan, Marcus Rohrbach, Jeff Donahue, Raymond Mooney, Trevor Darrell, and Kate Saenko. Sequence to sequence – video to text. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015a. 18, 25, 118

Subhashini Venugopalan, Huijuan Xu, Jeff Donahue, Marcus Rohrbach, Raymond Mooney, and Kate Saenko. Translating videos to natural language using deep recurrent neural networks. In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL), 2015b. 53, 119, 141

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In Proceedings of the International Conference on Machine Learning (ICML), 2008. 172

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. arXiv:1411.4555, 2014. 22, 53, 95, 119, 141

Adam Vogel and Dan Jurafsky. Learning to follow navigational directions. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2010. 105

Dong Wang and Xiaoyang Tan. Unsupervised feature learning with c-svddnet. Pattern Recognition, 2016. 48

Peng Wang, Qi Wu, Chunhua Shen, Anton van den Hengel, and Anthony Dick. Explicit knowledge-based reasoning for visual question answering. arXiv:1511.02570, 2015. 4, 165

Peng Wang, Qi Wu, Chunhua Shen, Anton van den Hengel, and Anthony Dick. Fvqa: Fact-based visual question answering. arXiv:1606.05433, 2016. 4

Peter Welinder and Pietro Perona. Online crowdsourcing: rating annotators and obtaining cost-effective labels. In CVPR Workshops, 2010. 94

Peter Welinder, Steve Branson, Pietro Perona, and Serge J Belongie. The multidimensional wisdom of crowds. In Advances in Neural Information Processing Systems (NIPS), 2010. 94

Jason Weston, Samy Bengio, and Nicolas Usunier. Wsabie: Scaling up to large vocabulary image annotation. In International Joint Conference on Artificial Intelligence (IJCAI), 2011. 96

Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. arXiv:1410.3916, 2014. 53, 119, 120, 142

Michael Wick, Andrew McCallum, and Gerome Miklau. Scalable probabilistic databases with factor graphs and mcmc. In Very Large Data Base (VLDB), 2010. 105, 108

Terry Winograd. Procedures as a representation for data in a computer program for understanding natural language. Technical report, DTIC Document, 1971. 1, 20, 35

Terry Winograd. Understanding natural language. Cognitive psychology, 1972. 19, 37, 52, 53, 54

Yuk Wah Wong and Raymond J Mooney. Learning for semantic parsing with statistical machine translation. In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL). Association for Computational Linguistics, 2006. 52

William A. Woods. Semantics and quantification in natural language question answering. Advances in computers, 1978. 52

Jiajun Wu, Ilker Yildirim, Joseph J Lim, Bill Freeman, and Josh Tenenbaum. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In Advances in Neural Information Processing Systems (NIPS), pages 127–135, 2015. 167

Qi Wu, Chunhua Shen, Anton van den Hengel, Peng Wang, and Anthony Dick. Image captioning and visual question answering based on attributes and their related external knowledge. arXiv:1603.02814, 2016a. 4

Qi Wu, Peng Wang, Chunhua Shen, Anton van den Hengel, and Anthony Dick. Ask Me Anything: Free-form Visual Question Answering Based on Knowledge from External Sources. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016b. 4, 58, 143, 155, 165

Zhibiao Wu and Martha Palmer. Verbs semantics and lexical selection. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 1994. 99, 100, 110, 124, 127

Caiming Xiong, Stephen Merity, and Richard Socher. Dynamic memory networks for visual and textual question answering. arXiv:1603.01417, 2016. 4, 58, 143, 155, 156

Huijuan Xu and Kate Saenko. Ask, attend and answer: Exploring question-guided spatial attention for visual question answering. arXiv:1511.05234, 2015. 5, 58, 143, 155, 167

Kelvin Xu, Jimmy Ba, Ryan Kiros, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. Proceedings of the International Conference on Machine Learning (ICML), 2015. 5, 22, 48, 58, 143, 167

J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009. 16, 26, 47, 48, 50, 62, 63, 67, 76

Jiaolong Yang, Peiran Ren, Dong Chen, Fang Wen, Hongdong Li, and Gang Hua. Neural aggregation network for video face recognition. arXiv:1603.05474, 2016. 167

Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. Stacked attention networks for image question answering. arXiv:1511.02274, 2015. 4, 58, 142, 143, 147, 148, 155, 156, 184, 248

Dingjun Yu, Hanli Wang, Peiqiu Chen, and Zhihua Wei. Mixed pooling for convolutional neural networks. In International Conference on Rough Sets and Knowledge Technology, pages 364–375. Springer, 2014. 48

Licheng Yu, Eunbyung Park, Alexander C. Berg, and Tamara L. Berg. Visual madlibs: Fill in the blank image generation and question answering. Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015a. 4, 120

Licheng Yu, Eunbyung Park, Alexander C Berg, and Tamara L Berg. Visual madlibs: Fill in the blank description generation and question answering. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), pages 2461–2469, 2015b. 8, 55, 144, 145

Lotfi A Zadeh. Fuzzy sets. Information and control, 1965. 110

Wojciech Zaremba and Ilya Sutskever. Learning to execute. arXiv preprint arXiv:1410.4615, 2014. 122

Matthew D Zeiler and Rob Fergus. Stochastic pooling for regularization of deep convolutional neural networks. International Conference on Learning Representations (ICLR), 2013. 5, 48

John M Zelle and Raymond J Mooney. Learning to parse database queries using inductive logic programming. In Proceedings of the Conference on Artificial Intelligence (AAAI), 1996. 37, 52, 98

Luke S Zettlemoyer and Michael Collins. Online learning of relaxed ccg grammars for parsing to logical form. In EMNLP-CoNLL, 2007. 20, 37, 52, 93, 97, 104

Luke S Zettlemoyer and Michael Collins. Learning context-dependent mappings from sentences to logical form. In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2, pages 976–984. Association for Computational Linguistics, 2009. 52

Bolei Zhou, Yuandong Tian, Sainbayar Sukhbaatar, Arthur Szlam, and Rob Fergus. Simple baseline for visual question answering. arXiv:1512.02167, 2015. 155

Linchao Zhu, Zhongwen Xu, Yi Yang, and Alexander G Hauptmann. Uncovering temporal context for video question and answering. arXiv:1511.04670, 2015. 4, 56, 145

Yuke Zhu, Oliver Groth, Michael Bernstein, and Li Fei-Fei. Visual7W: Grounded Question Answering in Images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016. 4, 55, 58, 143, 144, 164

C Lawrence Zitnick, Devi Parikh, and Lucy Vanderwende. Learning the visual interpretation of sentences. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2013. 53, 119, 141

List of Figures

List of Tables

Mateusz Malinowski

mateuszm@google.com

Objective

Building responsive machines that understand natural language, surrounding environment, as well as human intentions, all necessary for human-like communication.

Positions

2017–now

Research Scientist, *Google DeepMind*, London, U.K.
Description: Working towards holistic machines.
Webpage: mateuszmalinowski.github.io (Personal)

2011–2016

Research Student, *Max Planck Institute for Informatics*, Saarbrücken, Germany.
Group: Scalable Learning and Perception
Advisor: Dr. Mario Fritz
Description: Built the first dataset and architectures that answer questions about images. Conducted research on Deep Learning, Spatial Reasoning, and Retrieval.
Webpage: people.mpi-inf.mpg.de/~mmalinow

2010

Research Assistant, *Cluster of Excellence on Multimodal Computing and Interaction*, Saarbrücken, Germany.
Group: Probabilistic Machine Learning and Medical Image Processing
Advisor: Prof. Matthias Seeger

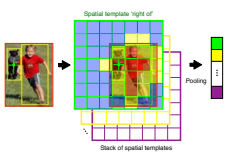
2009–2010

Research Assistant, *Max Planck Institute for Informatics*, Saarbrücken, Germany.
Group: High Dynamic Range Imaging and Perception Issues in Graphics
Advisor: Prof. Karol Myszkowski

Research Projects (Google Scholar profile)

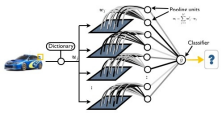
Towards a Visual Turing Challenge (NIPS, ICCV, ICMR, BMVC, IJCV), project page.

In this line of research, we build machines that answer questions about the content of images as well as we develop automatic performance metrics that monitor progress on this subjective task. We introduce the first dataset for the visual question answering task with about 1.5k real-world indoor images and 12.5k natural language questions. We also develop and investigate symbolic and neural approaches to handle the task. Both methods are trained only on image-question-answer triples. Moreover, our new performance metrics embrace word ambiguities and many interpretations of a question and a scene in benchmarking different architectures. This research is covered in Bloomberg Business.



Learning Spatial Relations, project page.

This project investigates a data-driven approach to learn spatial representation for image-to-text retrieval. For this purpose we have collected 53 structured queries that augment SUN09 dataset. The method improves over the state-of-the-art on the image-to-text retrieval task, and is very competitive to hand-engineered spatial features.



Learning Smooth Pooling Regions (BMVC), project page.
In this project, we argue for a data-driven approach to learn spatial pooling stage - an important part of the popular recognition architectures. Our formulation enables joint and discriminative training of the spatial pooling operator together with a classifier. The experimental evaluation shows that our approach significantly improves over similar recognition architectures with hand-designed spatial pooling stage.

Education

2011–2017 **PhD Student**, *Saarland University*, Saarbrücken, Germany.
Department of Computer Science
Thesis: Towards Holistic Machines:
From Visual Recognition To Question Answering About Real-World Images
Advisor: Dr. Mario Fritz
Committee: Prof. Trevor Darrell, Prof. Manfred Pinkal, Prof. Jens Dittrich, Dr. Qianru Sun

2009–2011 **Master of Science, Honor's degree**, *Saarland University*, Saarbrücken, Germany.
Department of Computer Science
Grade: Excellent, 1.3 in German Scale, 128 ECTS
Thesis: Optimization Algorithms in the Reconstruction of MR Images: A Comparative Study
Advisor: Prof. Matthias Seeger, Reviewer: Prof. Matthias Hein

2008–2009 **Erasmus Student**, *Saarland University*, Saarbrücken, Germany.
Department of Computer Science

2004–2009 **Undergraduate Studies**, *University of Wrocław*, Wrocław, Poland.
Department of Computer Science

Talks

2016 **Towards a Visual Turing Challenge**, *Microsoft Research*, Cambridge, U.K.

2016 **Towards a Visual Turing Challenge**, *DeepMind*, London, U.K.

2015 **Ask Your Neurons: A Neural-based Approach to Answering Questions about Images**, *ICCV*, Santiago, Chile.

Awards and Scholarships

2011 **Honor's degree in Computer Science**, *Saarland University*, Saarbrücken, Germany.

2010–2011 **International Max Planck Research School Scholarship**, *Saarbrücken*, Germany.

Academic and Working Experience

2015-2016 **Teaching Assistant**, *Deep Learning Seminar*, Max Planck Institute for Informatics, Saarbrücken.

2012-2013 **Teaching Assistant**, *Probabilistic Graphical Models and their Applications*, Max Planck Institute for Informatics, Saarbrücken.

2007–2008 **Programmer**, *USOS deployment*, University of Wrocław, Wrocław.
Advisor: Piotr Witkowski

2007 **Programmer**, *Wevo Developer*, University of Wrocław, Wrocław.
Advisors: Dr. Piotr Wnuk Lipiński, Marcin Brodziak

Academic Services

Advisor I helped in advising
- Ashkan Mokarian. Master's Thesis co-advisor (2016).
  Main supervisor: Dr. Mario Fritz
  Title: Deep Learning for Filling Blanks in Image Captions
- Sreyasi Nag Chowdhury. Master's Thesis co-advisor (2015).
  Main supervisors: Dr. Andreas Bulling, and Dr. Mario Fritz.
  Title: Contextual Media Retrieval Using Natural Language Queries
  Now, she is a PhD student in MPI for Informatics

Tutor 2nd Summer School on Integrating Vision and Language: Deep Learning, Malta, 2016

Reviewer (Journals)
Transactions on Pattern Analysis and Machine Intelligence (TPAMI)
International Journal of Computer Vision (IJCV)
Journal of Mathematical Imaging and Vision (JMIV)
Information Processing and Management (IPM)
Transactions on Computational Intelligence and AI in Games
Language and Linguistics Compass

Reviewer (Conferences)
International Conference on Computer Vision (ICCV)
Conference on Computer Vision and Pattern Recognition (CVPR)
Neural Information Processing Systems (NIPS)
European Conference on Computer Vision (ECCV)
Asian Conference on Computer Vision (ACCV)
The European Chapter of the ACL (EACL)
International Conference on Pattern Recognition (ICPR)

Participant
IEEE member, 2016
BMVA member, 2014, 2017
Deep Learning Reading Group, Saarbrücken, Germany, 2015 (Organizer)
GCPR R3 Session. Saarbrücken, Germany, 2013
Graduate Summer School: Deep Learning, Feature Learning. IPAM, UCLA, USA, 2012
Microsoft PhD Summer School. MSR, Cambridge, UK, 2012

Additional

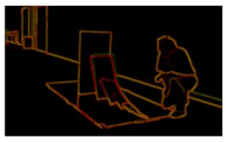
Languages Polish (native speaker), English (fluent)
German (basic), Russian (basic)

Online Courses (Coursera)
Startup Engineering, Neural Networks for Machine Learning
Modern & Contemporary American Poetry, Compilers

Publications (Google Scholar profile)

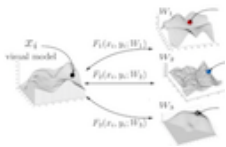
Towards a Visual Turing Challenge, project page.

- Mateusz Malinowski, Marcus Rohrbach, and Mario Fritz
  Ask Your Neurons: A Deep Learning Approach to Visual Question Answering
  International Journal of Computer Vision (**IJCV**), 2017 paper
- Adam Santoro, David Raposo, David G.T. Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, Timothy Lillicrap
  A simple neural network module for relational reasoning
  Technical Report, 2017 paper
- Mateusz Malinowski, and Mario Fritz
  Tutorial on Answering Questions about Images with Deep Learning
  2nd Summer School on Integrating Vision and Language: Deep Learning, *Malta, March 21-24*, 2016 paper
- Ashkan Mokarian, Mateusz Malinowski, and Mario Fritz
  Mean Box Pooling: A Rich Image Representation and Output Embedding for the Visual Madlibs Task
  British Machine Vision Conference (**BMVC**), *York, UK, September 19-22*, 2016 paper
- Sreyasi Nag Chowdhury, Mateusz Malinowski, Andreas Bulling, and Mario Fritz
  Contextual Media Retrieval Using Natural Language Queries
  ACM International Conference in Multimedia Retrieval (**ICMR**), New York, 2016 paper
- Mateusz Malinowski, Marcus Rohrbach, and Mario Fritz
  Ask Your Neurons: A Neural-based Approach to Answering Questions about Images
  IEEE International Conference on Computer Vision (**ICCV, Oral**), *Santiago, Chile, December 13-16*, 2015 paper
- Mateusz Malinowski and Mario Fritz
  A Multi-World Approach to Question Answering about Real-World Scenes based on Uncertain Input
  Neural Information Processing Systems (**NIPS**), *Montreal, CA, December 08-12*, 2014 paper
- Mateusz Malinowski and Mario Fritz
  Hard to Cheat: A Turing Test based on Answering Questions about Images
  Beyond the Turing Test (**AAAI Workshop**), *Austin, TX, January 25-26*, 2015 paper
- Mateusz Malinowski and Mario Fritz
  Towards a Visual Turing Challenge
  Learning Semantics (**NIPS Workshop**), *Montreal, CA, December 12*, 2014 paper



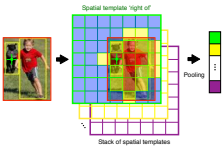
Intuitive Physics, .

- Apratim Bhattacharyya, Mateusz Malinowski, Bernt Schiele, and Mario Fritz
  Long-Term Image Boundary Extrapolation
  Technical Report, December, 2016 paper
- Apratim Bhattacharyya, Mateusz Malinowski, and Mario Fritz
  Long Term Boundary Extrapolation for Deterministic Motion
  NIPS Workshop on Intuitive Physics, December, 2016 paper



Zero-Shot Learning, project page.

- Zeynep Akata, Mateusz Malinowski, Mario Fritz, and Bernt Schiele
  Multi-Cue Zero-Shot Learning with Strong Supervision
  IEEE Computer Vision and Pattern Recognition (**CVPR, Spotlight**), June, 2016 paper



Spatial Relations in Retrieval, project page.

- Mateusz Malinowski and Mario Fritz
  A Pooling Approach to Modelling Spatial Relations for Image Retrieval and Annotation
  Technical Report, *Saarbrücken, Germany*, 2014 paper

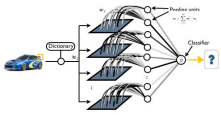
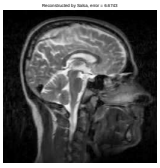


Image Recognition, project page.

- Mateusz Malinowski and Mario Fritz
  Learning Smooth Pooling Regions for Visual Recognition
  British Machine Vision Conference (**BMVC**), *Bristol, UK, September 09-13*, 2013 paper
- Mateusz Malinowski and Mario Fritz
  Learnable Pooling Regions for Image Classification
  International Conference on Learning Representations: Workshop Track (**ICLR Workshop**), *Scottsdale, Arizona, USA, May 02-04*, 2013 paper



Compressed Sensing.

- Mateusz Malinowski
  Optimization Algorithms in the Reconstruction of MR Images: A Comparative Study
  Master's Thesis, *Saarbrücken, Germany*, 2011 paper

Selected Publications

[1] *Ask Your Neurons: A Deep Learning Approach to Visual Question Answering*
Mateusz Malinowski, Marcus Rohrbach, Mario Fritz
IJCV, 2017

[2] *Tutorial on Answering Questions about Images with Deep Learning*
Mateusz Malinowski, Mario Fritz
2nd Summer School on Integrating Vision and Language: Deep Learning, Malta, 2016

[3] *Ask Your Neurons: A Neural-based Approach to Answering Questions about Images*
Mateusz Malinowski, Marcus Rohrbach, Mario Fritz
ICCV (Oral), Santiago, Chile, 2015

[4] *Hard to Cheat: A Turing Test based on Answering Questions about Images*
Mateusz Malinowski, Mario Fritz
AAAI Workshop on 'Beyond the Turing Test', Austin, USA, 2015

[5] *A Multi-World Approach to Question Answering about Real-World Scenes based on Uncertain Input*
Mateusz Malinowski, Mario Fritz
NIPS, Montreal, Canada, 2014

[6] *Towards a Visual Turing Challenge*
Mateusz Malinowski, Mario Fritz
NIPS Workshop on 'Learning Semantics', Montreal, Canada, 2014

[7] *A Pooling Approach to Modelling Spatial Relations for Image Retrieval and Annotation*
Mateusz Malinowski, Mario Fritz
Technical Report, Saarbrücken, Germany, 2014

[8] *Learning Smooth Pooling Regions for Visual Recognition*
Mateusz Malinowski, Mario Fritz
BMVC, Bristol, UK, 2013

[9] *Learnable Pooling Regions for Image Classification*
Mateusz Malinowski, Mario Fritz
ICLR Workshop, Scottsdale, USA, 2013