Characteristics and Commonalities

Differentially Describing Datasets with Insightful Patterns

A dissertation submitted towards the degree
Doctor of Natural Sciences
of the Faculty of Mathematics and Computer Science
of
Saarland University

by

SEBASTIAN DALLEIGER

Saarbrücken, 2023

Date of colloquium	16 November 2023
Dean of faculty	Prof. Dr. Jürgen Steimle
Chairperson	Prof. Dr. Sven Rahmann
Reporters	Prof. Dr. Gerhard Weikum
	Prof. Dr. Thomas Gärtner
	Prof. Dr. Jilles Vreeken
Academic Assistant	Dr. Philip Wellnitz

Zusammenfassung

Kern der empirischen Wissenschaft ist die Gewinnung von Erkenntnissen aus komplexen Daten. Durch den Aufstieg der computationellen Wissenschaft werden zunehmend zahlreichere, umfangreichere und reichhaltigere Datensätze verfügbar, mit deren Hilfe wir unser Wissen erweitern können. Gleichzeitig erschwert ein Mangel an geeigneten computationellen Werkzeugen die Analyse dieser Datensätze durch Domänenexperten. Insbesondere fehlt es an Methoden zur Identifizierung von *aufschlussreichen Mustern* (*insightful patterns*), d.h., Mengen von stark assoziierten Merkmalsausprägungen (*feature values*), die *informativ*, *kontrastierend*, *probabilistisch fundiert*, *statistisch fundiert* und durch *skalierbare* Algorithmen auffindbar sind. Diese Dissertation nutzt Ideen und Konzepte aus *Pattern-Set Mining*, *Maximum-Entropy Modeling*, statistischen Testverfahren und Matrixfaktorisierung, um Methoden zu entwickeln, die aufschlussreiche Muster identifizieren.

Abstract

Empirical science revolves around gaining insights from complex data. With the advent of computational science, increasingly more, larger, and richer datasets are becoming available to expand our scientific knowledge. However, the analysis of these datasets by domain experts is often impaired by a lack of suitable computational tools. In particular, there is a shortage of methods identifying *insightful patterns*, i.e., sets of strongly associated feature values that are *informative*, *contrasting*, *probabilistically sound*, *statistically sound*, and discoverable using *scalable* algorithms. This thesis leverages ideas and concepts from pattern-set mining, maximum-entropy modeling, statistical testing, and matrix factorization to develop methods for discovering insightful patterns.

Acknowledgments

I am deeply grateful to my PhD supervisor, Prof. Jilles Vreeken, for his invaluable guidance, encouragement, and support throughout my doctoral studies. His insightful feedback, expertise, and encouragement have been instrumental in shaping the directions of my research, helping me navigate the challenges of the PhD journey, and empowering me to become an independent researcher.

I also extend my gratitude to the members of my dissertation committee: Prof. Dr. Jilles Vreeken, Prof. Dr. Gerhard Weikum, and Prof. Dr. Thomas Gärtner, for their feedback, insights, and support. Their perspectives and diverse expertise have challenged me to think deeply and critically about my topics.

I am indebted to the scientists and colleagues who supported me generously with their time, valuable feedback, and whit whom I had many thought-provoking discussions. I am grateful to the many colleagues, friends, and family members who have supported me throughout this journey. Their encouragement, kindness, and unwavering belief in me have been a constant source of motivation and strength. Without their support, this research would not have been possible.

Lastly, I would like to express my thanks to the Saarland University, Max-Planck Institute for Informatics, and CISPA Helmholtz Center for Information Security for the resources, support, and intellectual community that this unique environment has provided, which have been central to my personal and academic development.

Contents

Characteristics and Commonalities

1
Introduction

Empirical science aims to understand how the world works based on observational and experimental data. One strategy to gain insights from such data is to discover sets of co-occurring feature values that are associated with an effect. In genetics, for example, sets of co-expressed genes might provide evidence for the presence of certain proteins that together could cause a disease (e.g., breast cancer), facilitating the development of targeted treatments [10, 12, 123]. And in neuroscience, sets of co-activated brain regions might point to potential peculiarities in neural wiring that could advance our understanding of healthy brain functioning as well as of neural disorders or diseases (e.g., autism or Alzheimer's) [14, 88, 178].

With the advent of computational science, increasingly more, larger, and richer datasets are becoming available to expand our scientific knowledge. However, the analysis of these datasets by domain experts is often impaired by a lack of suitable computational tools. In this thesis, we set out to develop such tools.

1.1 Motivation

As experts in algorithms and data analysis, computer scientists specializing in data mining or machine learning seem ideally positioned to design computational methods for discovering salient associations from complex data. While *supervised* machine learning is a powerful approach for tackling many prediction problems, *unsupervised* approaches are often better-suited for the exploratory settings we are

interested in, where the goal is to generate *understanding*. In particular, the task of identifying insightful sets of co-occurring feature values, called *patterns*, has been studied extensively in the data-mining subfield called *pattern mining* [1]. However, existing pattern mining methods fail to meet the requirements of domain scientists, as we elaborate below.

The classic motivating application of pattern mining methods, called *market-basket analysis*, is commercial, rather than scientific: Given a dataset of customer transactions in a store, where each transaction is a row (observation) and each purchasable item is a column (feature) of the data, identify items that are frequently bought together—e.g., to optimize product placement in physical stores or to recommend products in virtual stores, with the ultimate goal of increasing sales, profits, or customer satisfaction. Following the assumption that the more often we observe a pattern, the more interesting it is to the analyst, the goal here is to identify *frequent patterns*, i.e., sets of feature values that co-occur in a user-specified number or fraction of observations (absolute or relative co-occurrence frequency, respectively) [2, 71].

Traditionally, the goal of pattern mining algorithms has been to return *all* frequent patterns. As by definition, all subsets of a frequent pattern must also be frequent, these patterns can be mined very efficiently by gradually growing patterns from individual feature values [5]. However, the result is a highly redundant set of exponentially many patterns, only few of which are actually interesting to the analyst. To address the pattern explosion in frequent-pattern mining, *pattern-set mining* associates interestingness not with individual patterns but with *sets* of patterns. One influential approach here is to directly mine a concise set of non-redundant patterns that together describe the data well [68, 145]. Focusing on patterns that are *conjunctive*, i.e., that manifest as *positive* associations, we adopt this approach in our work, aiming to directly mine sets of patterns that are *informative*.

But while a single set of informative patterns summarizing a dataset could provide a basis for action in commercial settings, such a set does not generally suffice to advance our scientific knowledge.

Firstly, in scientific settings, subsets of observations (groups) often exhibit partially different feature-value distributions, corresponding to known or unknown covariates (e.g., biological sex, age, or medical conditions in the biomedical domain). This implies that *one* set of non-redundant patterns cannot describe the data succinctly. As traditional pattern miners ignore covariate groupings, they are hence prone to discovering spurious patterns that appear when considering all data at once but disappear when considering groups individually. Addressing this shortcoming of the status quo, we would like to identify individual sets of patterns per group in the data, rather than ignoring covariates. Our goal here is to discover a set of informative pattern sets highlighting (1) what is characteristic of a group, (2) what is common between groups, and (3) how to tell groups apart—regardless of whether the groups are known beforehand. That is, we seek to discover, both with and without prior knowledge of groups in the data, a set of informative pattern sets that is also *contrastive*.

Secondly, in scientific applications, the process by which we identify the pattern sets that describe a dataset matters. This diminishes the value of existing pattern-set miners, which are largely based on heuristics, and motivates us to develop methods that discover pattern sets in a theoretically well-founded way. More precisely, our goal is to model the data using the unique probability distribution that fits the data and does not incorporate any additional assumptions, i.e., the distribution with the maximum Shannon entropy [160] among all distributions fitting the data (maximum entropy distribution [85]). We then seek to mine patterns by identifying sets of features that are probabilistically independent of one another under this parsimonious distribution, thus ensuring that our pattern sets are *probabilistically sound*.

Thirdly, scientists are mostly dealing with noisy data, and hence, they require the pattern sets describing their data to be robust to minor fluctuations. This motivates us to use well-established model selection criteria (e.g., the Bayesian Information Criterion, BIC), statistical hypothesis testing under rigorous false discovery control (e.g., sequential false discovery rate adjustment), or a combination of the two (i.e.,

testing model selection criteria for significance) to ensure the statistical soundness of our results. Drawing inspiration from *statistically-significant pattern mining*, we thus aim to discover pattern sets that are *statistically sound*.

Finally, scientific data is often very large and high-dimensional, with many orders of magnitude more features (columns) than observations (rows). In this setting, pattern-set mining becomes computationally prohibitive, especially when we also ensure probabilistic soundness using maximum-entropy modeling and judge myriad individual patterns for statistical significance to ensure statistical soundness. To still gain insights from very large and high-dimensional scientific data, instead of deciding on individual patterns, we build and improve upon inherently scalable *matrix factorization* methods to model the data more efficiently. Treating the output of these methods as an assignment of patterns to data, we ensure that even for very large and very high-dimensional datasets, our pattern sets are discoverable using *scalable* algorithms.

In a nutshell: Motivated by the requirements of scientific applications, in this thesis, we set out to develop methods for discovering *insightful patterns*: sets of conjunctive patterns that are informative, contrasting, probabilistically sound, statistically sound, and discoverable using scalable algorithms. To this end, we draw on ideas from pattern-set mining, maximum-entropy modeling, statistical testing, and matrix factorization, as sketched in the following summary of our contributions.

1.2 Contributions

The chapters of this thesis have in common that they fulfill the requirements described in our motivation. Each chapter, however, focuses on different questions. In the following, we provide an overview of the questions asked in each chapter and how we answer them, thus outlining the remainder of this thesis.

Explainable Data Decompositions.

Pattern sets are easily interpretable models that are informative, even when considered in isolation. In the context of groups in data, they can explain differences and commonalities of such groups. Thus, they are contrasting, and even more informative in the biomedical domain. There, sets of feature interactions might explain deviating properties in a group (cancer) given the other group (healthy). Contrasts like these are thus highly informative to the researcher—if the groups are known beforehand. If no groups are known, however, there may still be groups in the data, corresponding to different data-generating processes that are potentially intertwined. Motivated by this scenario, in Chapter 2, we ask

> How can we identify *unknown* groups in *Boolean* data by leveraging their characteristic patterns?

Our goal here is to identify the unknown groups from the data. Similar to clustering or graph partitioning, we want to discover regions in the data that show significantly different distributions. Unlike these approaches, however, we not only contrast feature distributions (flat), but also contrast pattern distributions, thus also comparing many potentially higher-order interactions. We summarize this concept as the innovative grouping objective to *identify statistically significantly diverging pattern distributions*. This objective provides a rich, informative, and inherently interpretable test statistic, enabling us to statistically *guarantee* that each group is (tested-to-be) differently distributed from the others. Based on patterns, we can precisely and interpretably characterize *why* certain parts of the data constitute separate groups, we can explain *how* these groups are different from one another, and we can identify *which* properties the groups share among them.

Our approach to identifying statistically significantly diverging pattern distributions consists of two parts: Firstly, to parameterize groups with their pattern distribution, we formally introduce our algorithm Desc. Starting information-theoretically, we discuss a sub-modular optimization scheme from which we derive an efficient pattern-set mining algorithm, ensuring contrastiveness, probabilistic soundness,

and statistical soundness. Secondly, based on Desc's parameterization and our innovative grouping objective, we propose the Disc algorithm to discover high-quality groupings via an alternating optimization approach, enabling contrastiveness.

In brief, our main contributions here are:

1. We introduce the pattern distribution, which captures individual as well as higher-order interactions within one objective function.
2. We propose to group data into statistically significantly diverging clusters, thus guaranteeing contrastiveness.
3. We introduce an efficient search algorithm derived from our submodular optimization function.

Differentially Describing Groups of Graphs.

In Chapter 2, we considered tabular data, but not every dataset lends itself to a meaningful tabular representation. Notably, datasets designed to capture relationships between entities, such as interactions between proteins, are more naturally modeled as graphs (also called networks). In graphs, patterns are sets of edges that capture characteristic structure in the relationships observed, and as such, they are highly useful to the analyst. Thus, we ask:

> How can we describe *known* groups in *graph* data through their characteristic patterns?

Our setting is similar to that in Desc, with the difference that our input data are groups of graphs, rather than groups of rows in tables. More precisely, in Chapter 3, we are given a set of graphs and a partition of these graphs into groups, and our goal is to describe similarities and differences between graph groups by means of statistically significant subgraphs—a task we call *graph group analysis*.

To perform graph group analysis, we introduce the Gragra algorithm. Gragra combines maximum-entropy modeling with an information-theoretic model selection criterion and a statistical test to identify connected subgraphs that are significantly associated with individual groups.

In brief, our main contributions here are:

1. We propose graph group analysis as a task formalizing the discovery of informative and contrastive patterns in sets of graphs with known covariates.
2. We develop the theory to identify statistically significant subgraphs as patterns associated with one or more groups of graphs.
3. We use this theory to introduce an efficient algorithm solving the graph group analysis task.

Discovering Significant Patterns under Sequential False Discovery Control.

Although patterns convey insights into the data, they do not necessarily translate into scientific discoveries, as their feature interactions have to survive manual scientific scrutiny first. Since researchers are notoriously short on time, they should not be allocated tasks that are unlikely to convey useful knowledge, such as scrutinizing myriad patterns. As it is ultimately uncertain which patterns are worth their time, domain experts deem information criteria alone insufficient to make this decision. Rather, they require statistical certainty from rigorous hypothesis testing to make this call. The demand for statistical hypothesis testing is particularly high in the biomedical domain. Therefore, in Chapter 4, we ask

> How can we *sequentially* discover *pattern sets* such that each newly added pattern comes with *statistical guarantees*?

To answer this question while ensuring our general requirements, we introduce the first-of-its-kind framework for sequentially significant pattern-set mining. This framework combines the best of pattern-set mining with statistically significant pattern mining. Its novelty lies not only in this unique union, but also in further generalizations that allow us to do more than just discover contrasting and shared pattern-sets on arbitrarily many datasets. In particular, within this framework, we develop a new method to report those patterns that exhibit an empirical frequency that deviates significantly from our expectation. To this end, we *sequentially* control for false discoveries *during* the search (to avoid spurious results, yet achieve high statistical power), we

update our expectations whenever we discover a significant pattern (to avoid redundancy and achieve informativeness), we upper-bound the p-value computation using an easy-to-compute, yet accurate Chernoff (upper) bound (i.e., for efficiency), and we efficiently search for sets of significant patterns within the exponentially-sized search space.

In brief, our main contributions here are:

1. We introduce the new sequentially significant pattern-set mining problem.
2. We define a novel online search-aware Bonferroni correction targeting the family-wise error rate.
3. We propose to use an online sequential false discovery control targeting the false discovery rate.

The Relaxed Maximum Entropy Distribution.

The maximum entropy principle uniquely identifies the distribution that satisfies the observed pattern distributions but is otherwise maximally unbiased, thus ensuring our probabilistic soundness guarantees directly. Since inferring the maximum entropy distribution for pattern sets is exponentially complex in the number of patterns, exact inference quickly becomes intractable for all but trivial sets.

Prior work avoids the intractability of the inference problem by artificially prohibiting otherwise informative patterns whose inclusion would result in a high inference complexity. In Chapter 5, this motivates us to ask:

> How can we design a maximum entropy distribution that is both *unconstrained* and *efficiently inferable*?

Rather than *limiting* the distribution, we *relax* the distribution, introducing the novel *relaxed maximum entropy distribution* that permits efficient inference of unlimited distributions by dynamically factorizing the maximum entropy distribution into maximally entropic factors that we can learn from data. Formally, we show that the relaxed maximum entropy distribution is both PAC-learnable and consistent with standard maximum entropy.

In brief, our main contributions here are:

1. We define the new family of *relaxed* maximum entropy distributions.
2. We introduce an efficient member of this family.
3. We formally show that this distribution is PAC-learnable and consistent with standard maximum entropy.

Efficiently Factorizing Boolean Matrices.

If the dataset is reasonably large, pattern-set mining methods (such as the methods discussed in Chapter 2) provide a high degree of detail. That level of detail is achieved by relying on computationally costly search algorithms in an exponentially-sized search space. If the dataset is too large, they struggle to report meaningful results. In Chapter 6, we ask

> How can we discover groups in *Boolean* data and express them in terms of common concepts *at scale*?

This problem is commonly addressed using variants of matrix factorization, such as Non-Negative Matrix Factorization (NMF) or Principal Component Analysis (PCA), which achieve highly interpretable results—unless the data is Boolean. In the Boolean case, which is ubiquitous in the real world, the results returned by NMF are hard to interpret, because the input domain differs from the output domain. Addressing the interpretability problem of NMF on Boolean data, Boolean Matrix Factorization (BMF) uses Boolean algebra to decompose the input into low-rank Boolean factor matrices. These matrices are highly interpretable and very useful in practice, but they come at the high computational cost of solving an NP-hard combinatorial optimization problem.

To reduce the computational burden, we propose to relax BMF continuously using the innovative elastic-binary Elb regularizer, from which we derive a proximal gradient algorithm. Regularization alone, however, does not directly result in Boolean factors. Where prior work uses a constant regularization along with heavy and expensive post-processing, we propose to use a regularization rate, which achieves Boolean solutions without expensive post-processing.

In brief, our main contributions here are:

1. We introduce the novel elastic-net based elastic-binary Elb regularizer.
2. We define a regularization *rate* to achieve Boolean factors without post-processing.
3. We develop the Elbmf algorithm to efficiently factorize Boolean matrices.

1.3 Publications

The contributions laid out in this thesis were developed in a series of papers, as detailed below. The material presented in the following chapters is adapted from these papers. All our data, code, results, and additional details needed for reproducibility are publicly available, and we provide the links to the corresponding digital objects under the paper references below.

In all publications included in this thesis, the author of this thesis was involved as a first author, contributing to the main ideas, theory building, algorithm design, algorithm implementation, experimental evaluation, and preparation of the manuscript. For the contribution featured in Chapter 3, the first authorship was shared, and both authors contributed equally to the work, taking part in each of the previously mentioned steps.

Chapter 2

Sebastian Dalleiger and Jilles Vreeken. "Explainable Data Decompositions". In: *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 2020, pp. 3709–3716
Replication Material: doi.org/10.5281/zenodo.7548821

Chapter 3

Corinna Coupette, Sebastian Dalleiger, and Jilles Vreeken. "Differentially Describing Groups of Graphs". In: *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*. AAAI Press, 2022, pp. 3959–3967
Replication Material: doi.org/10.5281/zenodo.6342823

Chapter 4

Sebastian Dalleiger and Jilles Vreeken. "Discovering Significant Patterns under Sequential False Discovery Control". In: *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*. Ed. by Aidong Zhang and Huzefa Rangwala. ACM, 2022, pp. 263–272
Replication Material: doi.org/10.5281/zenodo.7548831

Chapter 5

Sebastian Dalleiger and Jilles Vreeken. "The Relaxed Maximum Entropy Distribution and its Application to Pattern Discovery". In: *20th IEEE International Conference on Data Mining, ICDM 2020, Sorrento, Italy, November 17-20, 2020*. Ed. by Claudia Plant et al. IEEE, 2020, pp. 978–983
Replication Material: doi.org/10.5281/zenodo.7548837

Chapter 6

Sebastian Dalleiger and Jilles Vreeken. "Efficiently Factorizing Boolean Matrices using Proximal Gradient Descent". In: *Thirty-Sixth Conference on Neural Information Processing Systems (NeurIPS)*. 2022, pp. 4736–4748
Replication Material: doi.org/10.5281/zenodo.7187021

2
Explainable Data Decompositions

In this chapter, we seek to discover and describe regions in *binary tabular data* that exhibit a characteristic pattern distribution. For example, given gene-expression data from cancer patients, we might seek to identify distinct cancer subtypes to facilitate the development of targeted treatments. Our goal is to discover the groups in such data, characterize *why* we deem these groups, explain *how* these groups are different from one another, and identify what properties they *share* among one another. Since we seek interpretable justifications for why we deem each region a group, we use sets of characteristic and common patterns. Rather than considering dense regions, as is common in clustering, we consider regions to be groups if they have statistically significantly different distributions, which we describe by means of insightful patterns that are informative for one or more groups.

We define the pattern composition problem in terms of a regularized maximum likelihood, in which we leverage the maximum entropy principle to model each group in the data using a concise set of patterns that characterizes it. As the search space of patterns and groups is large and unstructured, we propose the deterministic Disc and Desc algorithms, which together discover the pattern composition from data via an alternating-optimization approach. Empirical evaluation on synthetic and real-world data shows that Disc efficiently discovers

This chapter is based on the publication: Dalleiger and Vreeken [43].

groups and descriptions that accurately characterize the difference and the norm in easily understandable terms.

2.1 Introduction

Suppose we are analyzing gene expression data from cancer patients admitted to a hospital. Likely, our patients suffer from different (sub)types of cancers, each associated with its own characteristic genetic mutations. For example, breast cancer patients may exhibit mutations in BRCA1 or BRCA2 genes [151], while blood cancer patients may exhibit mutations in the CEBPA gene [137], and brain cancer patients may exhibit mutations in the IDH1 gene [13]. That is, the data consist of different *components*, i.e., parts that show significantly different pattern distributions.

Certain patterns may be characteristic of more than just one group; for example, a commonly mutated gene in *all* cancer patients is TP53 [143]. Therefore, the set of patterns that characterize the data can also be partitioned: Each such *pattern group* consists of those patterns that are characteristic of a distinct set of *data groups*. Together, the pattern groups give detailed yet easily interpretable insights into the justification of data groups, how they are different from one another, and what properties are shared among them.

Our goal is to jointly discover the groups of the data and those pattern groups that optimally characterize their similarities and differences—and we would like to do so both efficiently and in a statistically well-founded manner, where we only have to set a significance threshold α. We now seek to discover the *pattern composition* of a given binary tabular dataset. To this end, we model a group given a set of patterns using the Maximum Entropy principle [86]. That is, we use a maximum entropy estimator that satisfies the empirically observed frequencies of the given patterns but otherwise makes no further assumptions. We can then formulate the problem in terms of a likelihood maximization problem, where we are after that composition which achieves the highest overall likelihood. To avoid overfitting, we rely on the BIC model selection criterion. In other words, we are after the

most succinct way to summarize the data by partitioning it such that the parts exhibit significantly different distributions, and we would like to describe these distributions non-redundantly using only small and interpretable pattern sets.

Unfortunately, the search space for this problem is enormous: For a given dataset, there exists an exponential number of patterns, an exponential number of pattern sets, and an exponential number of partitions. Moreover, this space is not structured, barring efficient search for the optimum. We therefore introduce Disc, a deterministic method that heuristically discovers a good pattern composition. The main idea is that we split the problem into two parts and iterate between them until convergence. That is, for a given data decomposition, we propose to approximate the pattern groups using Desc, and then pass those pattern groups to Disc to discover refined data groups. In both steps, we rely on statistical tests to prune the search space and to ensure that we do not discover any spurious patterns or groups.

Through extensive experiments, we show that Desc and Disc work well in practice. Desc outperforms the state of the art in pattern-set mining, discovering succinct models with great efficiency, while Disc recovers meaningful groups. In case studies for which ground-truth data is available, we confirm that the groups and their characterizations make sense: The ecological niches and commonalities that Disc discovers correspond to the ground truth.

In summary, our contributions are

1. defining the *pattern composition* problem,
2. developing a fast method for discovering pattern groups,
3. introducing a fast method for discovering pattern compositions, and
4. validating both methods through extensive experiments on synthetic and real data.

The remainder of this chapter is structured as follows. After settling the preliminaries in Section 2.2, we introduce the maximum entropy distribution in detail in Section 2.3 (which we will refer back to throughout this thesis). We then state our problem formally in Section 2.4, and develop our algorithms, Desc and Disc, in Section 2.5.

Having discussed related work in Section 2.6, we empirically evaluate our methods in Section 2.7 before concluding the chapter in Section 2.8.

2.2 Preliminaries

We write 2^A for the powerset of any finite set A, and $\binom{A}{k}$ for the set of all subsets of A of size $k \in \mathbb{N}$. The set $A \bigtriangleup B$ is the symmetric difference of A and B, and we denote their disjoint union as $A \sqcup B$. For any $n \in \mathbb{N}$, we write $[n] = \{1, 2, \ldots, n\}$. The indicator function is $\mathbf{1}$. All logarithms are to base 2, and by convention, we use $0 \log 0 = 0$.

As data X, we consider multisets over d binary features $\mathcal{I}$, where each element $x \in X$ is independently drawn from the set of all possible elements $\Omega = 2^{\mathcal{I}}$. We write $\Pi \in \varpi(X)$ for a partitioning of data X into k non-empty pairwise disjoint subsets (classes), where $\Pi = \{X_1, \ldots, X_k\}$, such that

$$\bigcup_{X_j \in \Pi} X_j = X \ .$$

As patterns, we consider *itemsets* $x \subseteq \mathcal{I}$, and call $S \subseteq \Omega$ a *pattern set*, which is simply a set of patterns. We say that a data point $t \in X$ *supports* an itemset $x \in \mathcal{I}$ iff $x \subseteq t$, which, if counted, results in the empirical frequency

$$q_{X_i}(x) = |\{t \in X_i \mid x \subseteq t\}| \, / |X_i|$$

of x in X_i. We will maintain one pattern set S_i for each class X_i, and each pattern $x \in S_i$ is said to be *associated* with class X_i. Combined with empirical frequencies, a pattern set S_i is the sufficient statistic to define a probability distribution p over Ω.

2.3 The Maximum Entropy Distribution

First introduced by Jaynes in 1957 [85, 86], the principle of entropy maximization is intended for inferring probability distributions based on incomplete information. It does so by selecting the most *uninformed* probability distribution that satisfies all constraints given by the in-

formation available about the distribution. In other words, we choose the distribution that is least informative, subject to the known constraints. Originally, the principle is widely used to choose probability distributions in a variety of applications ranging from physics (e.g., in statistical thermodynamics [49] and quantum mechanics [18]), to economics (cf. [157]), to biology (e.g., to study protein-protein interactions [191], as the foundation of a precursor of Alpha Fold [120], and in computational enzyme design [196]). Due to its desirable property of shaping distributions without introducing a bias, the maximum entropy principle has also been applied in pattern mining, e.g., on binary tabular data [22, 194], real-valued data [91, 92], or networks [47, 165]. In the following, we motivate the maximum entropy distribution and introduce it succinctly.

The maximum entropy distribution is the distribution with the highest entropy subject to constraints imposed by available information. This principle in all its generality allows us to analytically derive a broad family of maximum entropy distributions from different kinds of constraints that we wish to impose (in addition to our restriction to probability distributions). It is common to do so algebraically in terms of its Lagrange relaxation of the linear program, which often results in many well-known probability distributions. Two canonical univariate examples are the *uniform distribution* (subject to no additional constraints) and the *normal distribution* (subject to constraints on mean and variance).

Our scenario, however, is more complicated: We are interested in distributions over *sets* of *discrete* random variables that maximize the entropy and satisfy the empirically observed frequencies of patterns and singletons (i.e., sets and elements) alike. To obtain this *discrete* and *multivariate* maximum entropy distribution that is constrained to match a given set of empirical frequencies, we consider distributions from the *polytope* of all feasible distributions,

$$\mathcal{P}_S^X \equiv \left\{ f \in \Omega \to [0,1] \,\middle|\, \mathbb{E}_f[x] = q_{X_i}(x)\ \forall x \in S,\ \textstyle\sum_t f(t) = 1 \right\} ,$$

which contains all, infinitely many distributions consistent with the empirical frequency q of patterns $x \in S$ in X_i.

However, not all distributions in $\mathcal{P}_S^X$ suit our needs: We need a distribution that does not introduce additional assumptions beyond the information that S specifies. From an information-theoretic point of view, additional assumptions correspond to additional information. We can measure the amount of information in a distribution using Shannon entropy,

$$H(p) = -\sum_x p(x) \log p(x) \ .$$

The lower the information content of a distribution p, the higher its Shannon entropy. We want to identify the feasible distribution that makes the fewest additional assumptions as the one with the highest entropy, i.e.,

$$f \equiv \operatorname*{arg\,max}_{f \in \mathcal{P}_S^X} \ H(f) \ , \tag{2.3.1}$$

which is the formalization of the principle of maximum entropy [86].

As our constraints are linear—as in the univariate case—we can easily state its Lagrangian [39]

$$H(p) - \sum_i \theta_i (p(x_i) - q(x_i)) - \theta_0 \left(\sum_i p(x_i) - 1 \right) \ .$$

Simplifying its derivative algebraically yields the general exponential model as our family of maximum entropy distributions. That is, given that the constraints of $\mathcal{P}_S^X$ are linear, the distribution p over transactions $t \in \Omega$ takes an exponential form

$$f(t) = f(t \mid S) = \theta_0 \prod_{x_i \in S} \theta_i^{\mathbf{1}[x_i \subseteq t]} \ , \tag{2.3.2}$$

where $\mathbf{1}$ is the indicator function, for appropriately chosen coefficients $\theta \in \mathbb{R}^{|S|+1}$ [39]. There are countless ways to estimate the globally optimal coefficients $\theta \in \mathbb{R}^{|S|+1}$ of this convex function, for example using gradient descent or Newton methods such as L-BFGS. We choose the specialized coordinate-descent optimization algorithm called *generalized iterative scaling* (GIS) [45], which was developed particularly

Algorithm 2.1: **Iterative Scaling**

Input: Itemsets $S \subseteq \Omega$, empirical frequencies $q \in S \to [0, 1]$
Output: Maximum entropy distribution p satisfying q

Start with arbitrary $\theta_i \; \forall x_i \in S$
Compute normalization constant θ_0
while has not converged
    for $x_i \in S$
        $\mu_i \leftarrow \mathbb{E}_f[x_i] = \sum_{\substack{y \in \Omega \\ x_i \subseteq y}} f(y \mid S)$
        $\nu_i \leftarrow q(x_i)$
        $\theta_i \leftarrow \theta_i \cdot \dfrac{\nu_i}{\mu_i}$
return θ

for estimating such coefficients for maximum entropy distributions. We summarize generalized iterative scaling as pseudocode in Algorithm 2.1.

As in the univariate case, we constrain the expectation of our distribution. Unlike in the univariate case, however, inferring the multivariate expectation

$$\mathbb{E}_f[x] = \sum_{y \in \Omega} f(y \mid S) \, \mathbf{1}\{x \subseteq y\} \; , \tag{2.3.3}$$

is defined over the exponentially-sized universe Ω containing all *possible combinations* of elements from $\mathcal{I}$. Inferring the expectation quickly becomes computationally intractable as the number of unique items in $\mathcal{I}$ grows, if done naïvely, and unless we take extra care. To make this inference tractable in practice, we use the following two key computational tricks.

The first trick is to factorize the distribution p_S over independent subsets of S [118].

Example 2.1. As an example, suppose that our dataset X is over items $\mathcal{I} = \{a, b, c, d, e\}$. Without the trick above, we have to sum over $|\Omega_{\mathcal{I}}| = 2^{\mathcal{I}} = 32$ transactions, regardless of S. Now suppose that $S = \{ab, cd, de\}$. Assuming a pairwise disjoint partitioning of S into $\bar{S}^1 = \{ab\}$ and $\bar{S}^2 = \{cd, de\}$, we can factorize p_S as $p_{\bar{S}}(x) = p_{\bar{S}^1}(x'_1) \times$

$p_{\bar{S}^2}(x'_2)$. To infer this factorized distribution, we only need to consider $|\Omega_{ab}| + |\Omega_{cde}| = 12$ transactions. Moreover, we can skip any factor distributions $p_{\bar{S}^i}$ that are irrelevant for inferring x (when $x'_i = \emptyset$). In the above example, for $x = ce$, we only have to infer $p_{\bar{S}^2}$, as $p_{\bar{S}^1}$ can be ignored.

That is, we partition S into pairwise disjoint subsets $\bar{S}^1, \ldots, \bar{S}^k$ of S that contain statistically independent pattern sets such that $S = \bigsqcup \bar{S}^i$. As the subsets are statistically independent, we can factorize the expectation $p_S(x) = \prod_i p_{\bar{S}^i}(x'_i)$ into the product of independent expectations, where we infer each term $p_{\bar{S}^i}$ for the $x'_i = t \cap (\bigcup_{x \in \bar{S}^i} x)$ subset of x which is covered by elements from $\bar{S}^i$.

This factorization simplifies not only the inference of p, but it also reduces the number of coefficients θ per factor, thus shrinking the size of the convex problems drastically. As an additional benefit, factorizing thus immediately addresses the relatively slow convergence rate of GIS when the problems are sufficiently large.

The second trick partitions every Ω into sets of *equivalent* itemsets, where a pair $u, v \in \Omega$ is equivalent if $p(u) = p(v)$. For each class of equivalent itemsets, we only have to infer the maximum entropy probability $f_i(x)$ once, and then we can scale the result by the size of the equivalence class. The computation of equivalence classes can be done efficiently in practice [118].

2.4 The Pattern Composition Problem

Having introduced the maximum entropy distribution as our workhorse, we are now ready to define our problem. Starting informally, our goal is to discover a partitioning Π of X by its characteristic pattern distributions, whose characteristics and commonalities we seek to identify in terms of informative patterns. That is, we aim to decompose the dataset X into disjoint subsets $X_1, \ldots, X_k$, such that every subset $X_j \subseteq X$ has a significantly different pattern distribution p_j, which we characterize in terms of informative patterns.

We achieve this goal in two steps. Firstly, for a given partitioning Π, we work under the assumption that partitions are pairwise indepen-

dent. This allows us to introduce a distribution p_j for each partition $X_j \in \Pi$. Using the maximum entropy principle for our distributions, we need to select their constraints in the form of patterns $S_j \subseteq \Omega$ and their frequencies q_j in partition X_j. In brief, for a given partitioning Π, we want to identify a succinct set $S_j \subseteq \Omega$ of patterns per part. Secondly, given a set of patterns $S \subseteq \Omega$, we seek to identify the partition $\Pi \in \varpi(X)$ such that each p_j has a characteristic distribution, given patterns in S.

As neither S nor Π is given to us for free, we jointly seek a decomposition $\Pi \in \varpi(X)$ of X, a succinct, non-redundant set of patterns S, and an assignment matrix A that associates patterns to groups in Π, such that we maximize the regularized likelihood

$$\ell(\Pi, S, A) = -\sum_{X_j \in \Pi} \log p(X_j \mid S_j) + r(\Pi, S, A) \ .$$

Here, $S_j \subseteq S$ is the subset of S that A indicates to be relevant for X_j, and $r(\Pi, S, A)$ is a regularization term that steers the problem away from trivial solutions, such as decomposing the data into singleton groups or including every possible pattern.

Regularization for Informative Patterns Our goal is to discover the maximally succinct, maximally non-redundant pattern set S of characteristic patterns. We say that a pattern x is *informative* for X with respect to S if we see a *significant* increase in likelihood if we include x in S. To determine whether a pattern is *informative* about X, we use a model selection criterion. Here, we opt for BIC as it is simple, efficiently computable, and—as we will see—works well in practice [158]. For a single group, we have $|S|$ degrees of freedom (df), and hence,

$$r(S) = \frac{1}{2} |S| \log |X| \ .$$

It is straightforward to generalize the above to multiple groups. Given a partitioning Π with k groups and a pattern set S, we need to determine which patterns $x \in S$ are *characteristic* for which group $X_i \in \Pi$. This is what our assignment matrix $A \in \{0, 1\}^{|S| \times |\Pi|}$ is for. It is a

binary matrix over groups and patterns, where $A_{ij} = 1$ if pattern $x_i \in S$ is informative for group X_j. The set of patterns that are informative for a group X_j is defined as $S_j = \{x_i \in S \mid A_{ij} = 1\}$. If a pattern is informative for multiple groups, we call it *common* or *shared* among those groups. For multiple groups, we need account for the assignment matrix A ($|S| \cdot |\Pi|$ df), the coefficients θ for $|\Pi|$ distributions ($[|S| + |\mathcal{I}|] \cdot |\Pi|$ df), and the partitioning as label per data point ($|X|$ df, constant), which amounts to

$$r(\Pi, S, A) = \frac{1}{2}\left[|\Pi| \cdot (2\,|S| + |\mathcal{I}|) + |X|\right] \log |X| \ ,$$

as our BIC cost r for multiple groups. With the BIC score now fully defined, we combine the above into our the problem statement.

Problem 2.1 (The Pattern Composition Problem). Given a transactional dataset X over items $\mathcal{I}$, our goal is to jointly discover

1. the partitioning $\Pi \in \varpi(X)$ of X into the fewest parts,
2. the smallest pattern set $S \subseteq \Omega$, and
3. the assignment matrix $A \in \{0,1\}^{|S| \times |\Pi|}$

such that

$$\ell(\Pi, S, A) = -\sum_{X_j \in \Pi} \log p(X_j \mid S_j) + r(\Pi, S, A) \ ,$$

is minimal.

Unsurprisingly, this is a difficult problem with a very large search space. Firstly, there exists a Bell number $B_{|X|}$ of possible partitionings Π. Secondly, the number of possible pattern sets is doubly exponential in the number of unique items in X, as $S \in 2^{2^{\mathcal{I}}}$. Finally, the joint objective does not exhibit structure that we can exploit directly for an efficient search, i.e., it is neither (anti-) monotone nor submodular. This raises the question: Can we still efficiently discover high-quality solutions in practice? The answer is affirmative—as we shall demonstrate in the following.

2.5 Algorithms

To efficiently discover good solutions to the pattern composition problem in practice, we separate the problem into two parts and take an alternating optimization approach. That is, starting from a partitioning π_0 in which all of X is in one part, we iterate between the following two steps until convergence. First, given a partitioning Π, we efficiently discover a high-quality pattern set S and assignment matrix A. Second, given a pattern set S, assignment matrix A, and partitioning Π, we discover a refined partitioning Π' that improves the value of our objective function.

Below, we discuss each of these steps in turn.

2.5.1 Discovering Patterns given a Partitioning

The first problem we consider is that of discovering a high-quality set of informative patterns $S \in 2^{\Omega}$ and assignment matrix A for a given partitioning $\Pi \in \varpi(X)$. Like our overarching problem, this problem is also too hard to solve exactly: There exist doubly exponentially many pattern sets, and our hard-to-compute score does not show any structure we can exploit. We hence choose to approximate the optimal result through an iterative greedy approach.

For a single dataset, Mampaey, Vreeken, and Tatti [118] showed that finding the set S with minimal ℓ is equivalent to finding the set that has the highest gain in likelihood in comparison to the empty pattern set $S_\emptyset$. That is, minimizing

$$\ell_j(S) = -\sum_{x \in X_j} \log p_j(x)$$

is equivalent to maximizing the information divergence

$$\arg\max_S \; D\left(p_S \parallel p_\emptyset\right) \tag{2.5.1}$$

measured as the Kullback-Leibler (KL) divergence

$$D(q\|p) = \sum_x q_x \log(q_x/p_x) \ . \tag{2.5.2}$$

This allows us to obtain a greedy solution to Eq. (2.5.1), which is equivalent to iteratively minimizing $\ell_j(S)$ directly. Our algorithm iteratively selects a candidate $x \in F$ from the set of candidates F with the highest marginal gain. To ensure that this procedure (under idealized circumstances) produces a result that is close to the optimum, we bound the error of our greedy solution.

Lemma 2.1. Eq. (2.5.1) is a *Submodular Function Maximization* problem. The greedy solution S is in the e^{-s/s^*}-radius of the optimal solution S^*, where $s = |S|$ and $s^* = |S^*|$.

To prove this, we use the framework of submodular function optimization, beginning with Theorem 6.1 from Mampaey, Vreeken, and Tatti [118].

Theorem 2.1 (Theorem 6.1 [118]). For the given consistent distribution q which has same support as p_S, the following holds true.

$$\arg\min_{x\in\Omega} \ell(X \mid S_i \cup \{x\}) = \arg\max_{x\in\Omega} D\left(p_{S_i\cup x} \parallel q\right) \ ,$$

In other words, the candidate with the highest marginal likelihood gain is identical to the candidate with the highest divergence,

$$f(S) = D\left(p_S \parallel q\right) \ .$$

This function is monotonic and submodular.

Proof (Monotonicity). We write the polytope of feasible distributions p as

$$\mathcal{P}_S \equiv \{p \in \Omega \to [0,1] | \sum p = 1, p_x = q_x \forall x \in S\} \ .$$

By consistency of q, we know that $\mathcal{P}_{S\cup\{x\}} \subseteq \mathcal{P}_S$. By tightening the constraints around p for *any* $x \in 2^T$, we are reducing the distance between any $p \in \mathcal{P}_S$ and q, and hence $f(S) \geq f(S \cup \{x\})$. □

Note that f is not necessarily strictly monotonic, since x might not carry additional information at all, and hence, $\mathcal{P}_{S\cup\{x\}} = \mathcal{P}_S$. However, this is not a problem, and in practice, we would not consider adding these uninformative candidates to our solution anyway.

Proof (Submodularity). A function f is a submodular set function if for all $S \subseteq T \subseteq 2^T$, it holds that

$$f(S \cup \{x\}) - f(S) \geq f(T \cup \{x\}) - f(T) \,,$$

or equivalently [94], for all $S, T \subseteq 2^T$,

$$f(S \cap T) + f(S \cup T) \leq f(S) + f(T) \,.$$

With this definition, we can derive that the Kullback-Leibler divergence, which defines our objective function, is submodular:

$$\begin{aligned} f(S \cup T) + f(S \cap T) &= D(p_{S\cup T} \parallel q) + D(p_{S\cap T} \parallel q) \\ &\leq D(p_S \parallel q) + D(p_T \parallel q) \\ &\qquad - D(p_{S\cap T} \parallel q) + D(p_{S\cap T} \parallel q) \\ &= D(p_S \parallel q) + D(p_T \parallel q) \\ &= f(S) + f(T) \,. \end{aligned}$$

Here, we factorized the divergence between $p_{S\cup T}$ and q into the divergences of p_S and p_T and subtracted the divergence to the distribution $p_{S\cap T}$ concerned with the intersection. □

Proof (Quality of Greedy Approximation). Observe that inserting patterns S_i into the *optimal* sufficient statistics S^* does *not* gain information, i.e., $f(S^*) \leq f(S^* \cup S_i)$. Therefore, we can simplify the term $f(S^* \cup S_i)$. For this, we expand the right-hand side by a sequence of terms excluding iteratively more optimal elements from S^*. That is, letting $k = s^* = |S^*|$,

we consider

$$
\begin{aligned}
f(S^* \cup S_i) + \underline{f(S_i \cup S^* \setminus \emptyset)} - f(S_i \cup S^* \setminus \emptyset) \\
+ f(S_i \cup S^* \setminus \{x_k^*\}) - \underline{f(S_i \cup S^* \setminus \{x_k^*\})} \\
+ f(S_i \cup S^* \setminus \{x_{k-1}^*, x_k^*\}) - f(S_i \cup S^* \setminus \{x_{k-1}^*, x_k^*\}) \\
\dots \\
+ f(S_i \cup S^* \setminus S^*) - f(S_i \cup S^* \setminus S^*)
\end{aligned}
$$

until we reach the final term $f(S_i) = f(S_i \cup (S^* \setminus S^*))$ of the empty set. In other words, we create a sequence of canceling term pairs by shrinking S^* until it is empty. Next we simplify this expansion by reordering terms, in such a way that neighbors differ by one element (see underlining). After reordering the terms, and defining $S^*_{\ominus j} = S^* \setminus \{x_1^* \dots x_j^*\}$, we obtain

$$
\begin{aligned}
f(S^* \cup S_i) &= f(S^* \cup S_i) - f(S^* \cup S_i \setminus \emptyset) + f(S_i \cup (S^* \setminus S^*)) \\
&\quad + \sum_j f(S_i \cup S^*_{\ominus j-1}) - f(S_i \cup S^*_{\ominus j}) \\
&= f(S_i) + \sum_j f(S_i \cup S^*_{\ominus j-1}) - f(S_i \cup S^*_{\ominus j}) \,.
\end{aligned}
$$

Using f's submodularity, we bound the residuals $f(S_i \cup S^*_{\ominus j-1}) < f(S_i \cup \{x_j^*\})$ and $f(S_i \cup S^*_{\ominus j}) < f(S_i)$ to then obtain

$$
\begin{aligned}
f(S^*) &\leq f(S_i) + \sum_j f(S_i \cup S^*_{\ominus j}) - f(S_i \cup S^*_{\ominus j-1}) \\
&\leq f(S_i) + \sum_j f(S_i \cup \{x_j^*\}) - f(S_i) \,.
\end{aligned}
$$

In the ith iteration, we denote the candidate with the highest marginal gain as $\hat{x}_i$, and we define $S_{i+1} = S_i \cup \{\hat{x}_i\}$. Bounding $f(S_i \cup \{x_j^*\}) \leq$

$f(S_{i+1})$, we simplify further as

$$\begin{aligned} f(S^*) &\leq f(S_i) + \sum_{j \in [k]} f(S_i \cup \{x_j^*\}) - f(S_i) \\ &\leq f(S_i) + \sum_{j \in [k]} f(S_{i+1}) - f(S_i) \\ &= f(S_i) + k \cdot (f(S_{i+1}) - f(S_i)) \ . \end{aligned}$$

Now combining the left-hand side and the right-hand side, we get the following simple bound on the optimal solution:

$$f(S^*) \leq f(S_i) + k \cdot (f(S_{i+1}) - f(S_i)) \ ,$$

for which a sequence of equivalences leads step-by-step to the bound stated in Lemma 2.1:

$$\begin{aligned} & 1/k \, (f(S^*) - f(S_i)) \leq f(S_{i+1}) - f(S_i) \\ \Leftrightarrow \ & 1/k f(S^*) + (1 - 1/k) f(S_i) \leq f(S_{i+1}) \\ \Leftrightarrow \ & (1 - 1/k) f(S_i) \leq f(S_{i+1}) - f(S^*) + (1 - 1/k) f(S^*) \\ \Leftrightarrow \ & f(S^*) - f(S_{i+1}) \leq (1 - 1/k)(f(S^*) - f(S_i)) \\ \Leftrightarrow \ & g_{i+1} \leq (1 - 1/k) g_i \ . \end{aligned}$$

By induction, for step l, we bound $g_l \leq (1 - 1/k)^l g_0$. Using $(1 + x) \leq e^x$, we derive $g_l \leq e^{-l/k} g_0$, which further simplifies to

$$f(S^*) - f(S_l) \leq e^{-l/k} f(S^*) \ ,$$

as $g_0 \leq f(S^*)$. In other words, the solution S_l lies within the $e^{-l/k}$ radius of $f(S^*)$, such that denoting $k = |S^*| = s^*$ and $l = |S| = s$ for $S = S_l$, we obtain the bound as stated in Lemma 2.1. By setting $l = k$, for example, our solution lies within the $1/e \approx 0.368$ radius of S^*. Setting $l = 2k$, however, our solution lies within the 0.135 radius. For our practical purposes, this bound is good enough. □

Summarizing the above, we now know that maximizing the divergence D is a submodular function maximization problem that can

be solved with approximation guarantees, and that the estimates of greedily optimizing ℓ and D are equivalent.

However, the greedy algorithm to minimize Eq. (2.5.1) is still prohibitively slow in practice: It repeatedly has to evaluate the Kullback-Leibler divergences $D(p_S \parallel p_{S\cup\{x\}})$ to measure the information gain of adding an itemset x to S, and this computation relies on the computationally costly inference of frequencies using p. Nonetheless, this formulation *does* allow us to derive a computationally efficient admissible heuristic.

To reduce the complexity of computing D, we want to reduce the number of queries it makes to p. In its full computation, it considers the frequencies of both pattern x itself and its exponentially many subsets $y \subset x$. Ignoring these subsets permits the lower bound

$$h(x \mid S) = nq(x) \log q(x)/p_S(x) \ ,$$

where we use the fact that $p(x \mid S)$ is equivalent to $q(x)$ for $x \in S$.

Interpreting this score, we see that h favors patterns with the following three properties. Firstly, h favors sufficiently frequent patterns (i.e., with high $q(x)$). Secondly, h favors insufficiently well-explained patterns (i.e., with high difference $q(x) - p_S(x)$). That is, because h is 'large' whenever $q(x) \gg p_S(x)$, we favor candidates that increase our likelihood significantly. Upon closer inspection, we notice that this part of h essentially decides whether

$$q(x) \not\approx p_S(x)$$

in terms of information gain under our BIC cost. In other words, we see that h probabilistically soundly decides on the independence between x and elements in S via $q(x)$ and $p_S(x)$, if p is faithful to the available information in q_S. For example, if this holds, then h decides whether $q(ab) \approx q(a)q(b)$ or not, for any two elements $a \neq n \in \mathcal{I}$. To ensure this probabilistic interpretation, we need to provide our distributions p_j with the available information for each element in $\mathcal{I}$, or in other words, the marginal distribution of each item. Combined with the maximum entropy property of $q(x) = p_S(x)\ \forall x \in S$, this ensures probabilistically

sound decision-making using h. In practice, it suffices to include all elements from $\mathcal{I}$ in each S_j implicitly.

Due to the decomposition of the data, this admissible heuristic easily generalizes to an admissible normalized *lower-bound information gain* over multiple groups $h(x)$ as $\mathbb{E}_{S_j}\left[h(x \mid S_j)\right]$. In general, our pattern-set discovery strategy is hence as follows. In the current iteration i, the last pattern set S^{i-1} is known and fixed. We use h to select that itemset x in the set of candidates $F \subseteq \Omega$ which has the highest marginal gain. That is, until convergence of ℓ, we iterate

$$S^i \leftarrow S^{i-1} \cup \arg\max_{x \in F} h(x \mid S^{i-1}) \ .$$

This leaves us to specify the candidate set F. Naïvely, we could set $F = \Omega$. However, this is not practical: Ω is typically prohibitively large, and it contains exponentially many candidates that will be uninformative with regard to patterns in S. To cope with this search space, we propose a more effective search strategy, that takes into account what S can already explain well. In a nutshell, we iteratively generate candidates by merging pairs of patterns $x, y \in S \cup \mathcal{I}$ into a candidate $x \cup y \in F$. Because F contains candidates with more than one element, we need to ensure that our heuristic h can (at least) probabilistically soundly assess dependencies between pairs of singletons. To do so, we need to make available the marginal distributions for each singleton $\mathcal{I}$ to each p_j, i.e., we need to provide singletons $\mathcal{I}$ to each set $S_j = \mathcal{I}$ for every $X_j \in \Pi$.

However, we only want to consider the subset of candidates that will surely reduce our objective ℓ. Those are candidates $z \in F$ for which $h(z \mid S) > r(z)$, where $r(z) = r(\Pi, S \cup \{x\}) - r(\Pi, S)$. Similarly, we assign a candidate i to a group j if it yields a gain in ℓ_j, i.e.,

$$A_{ij} = 1 \iff r_j(x_i) < h(x_i \mid S_j) \ , \tag{2.5.3}$$

where the cost $r_j(z)$ is $r(\Pi, S, A') - r(\Pi, S, A)$. Here A' is equivalent to A, but with $A_{ij} = 1$.

Putting the above together, we have algorithm DESC, whose pseudocode we give as Algorithm 2.2. In short, starting with the singleton-

Algorithm 2.2: **Desc** for Characterizing Groups

Input: Data X, partitioning Π
Output: Distributions p_j, pattern set S, assignment A

```
1  S ← {x ∈ I}
2  p ← infer p(· | S_j) for each group X_j
3  F ← {z = x ∪ y | x, y ∈ S, r(z) < h(z)}
4  while F ≠ ∅
5      z ← arg max_{x∈F} h(x)
6      A' ← according Eq. (2.5.3) wrt z
7      S' ← S ∪ {z} if z assigned to a group
8      p' ← infer p(· | S'_j) for each group X_j
9      if ℓ(Π, S', A') < ℓ(Π, S, A)
10         A ← A'; S ← S'; p ← p'
11         F ← {z = x ∪ y | x, y ∈ S, r(z) < h(z)}
12     else
13         F ← F \ {z}
14 return (p, S, A)
```

only model (ll. 1–3), we generate our initial batch of candidates F (l. 4). We consider these candidates in descending order of h (l. 6) and evaluate each $z \in F$ (l. 7–9). If the objective improves, we keep the candidate (ll. 11–12)—otherwise, we reject it (l. 13).

The computational complexity of Desc depends on the number of candidates in F, which is quadratic in the number of patterns in S, and can grow up to $|\Omega|$. In practice, however, the properties of the maximum entropy distribution together with the BIC regularizer keep the size of S small, in the order of tens to hundreds of patterns, say $S_{\max}$. The worst-case complexity of Desc is dominated by the inference of the distributions p_j, and is hence in PP. The average complexity γ of p is much lower [118], however, in that the average complexity of Desc is $O(\gamma \cdot |S_{\max}|^2)$.

2.5.2 Discovering the Composition

Next, we consider the orthogonal problem of discovering a high-quality partitioning Π given a pattern set S and assignment matrix A. As there is no effective exact search for the optimal partitioning, we

again rely on heuristics. In particular, we take a top-down approach, where we iteratively refine the current partitioning Π using the patterns in S.

Our strategy is based on the idea that significantly different distributions of patterns are an indicator for the presence of latent factors of unknown groups. In other words, we say that a group was generated using a latent data source that left a distinctly distributed fingerprint of patterns in the data. By narrowing down a given group to a subset with a distribution that stands out from the rest of the data, we can refine the current partitioning to identify these latent parts as separate groups. We can also use this observation in reverse: When we narrow down a group and find that the pattern distribution we so obtain is not significantly different from the remainder or the other groups, we do not want this candidate group to be part of our solution.

We write p_j^x for the pattern distribution we infer on that part of group X_j where pattern x occurs. Likewise, we consider $p_j^{\not x}$ over that part of X_j where x does not occur. We measure the divergence between two distributions with the same support using the Jensen-Shannon divergence

$$JS(P, Q) = D(P \parallel M) + D(Q \parallel M) \ ,$$

where $M = (P + Q)/2$. The scaled $JS(p_j^x, p_j^{\not x})$ statistic is asymptotically χ^2 distributed with $|S| - 1$ df [124]. From this, we get a p-value for a single test. However, as we test many hypotheses, i.e., candidate refinements, we need to control for multiple hypothesis testing. Hence, we correct for the family-wise error rate (FWER) by adjusting the significance level α using Bonferroni correction [25].

For a given S and for any partitioning $\Pi \in \varpi(X)$, we write $A(\Pi, S)$ as the assignment matrix that characterizes the partitioning Π with S and minimizes our objective function with respect to Eq. (2.5.3). Formally, for a given S, the problem of discovering groups is

$$\begin{aligned} \underset{\Pi \in \varpi(X)}{\arg\min} \quad & \ell(\Pi, S, A(\Pi, S)) \\ \text{subject to} \quad & p_i, p_j \text{ significantly JS-divergent } \forall i \neq j \end{aligned}$$

This is, again, a hard problem, and again, the search space is large and unstructured. We therefore employ a greedy top-down approach. Starting with a single group $X_j \in \Pi$, we decompose X_j into two subset X_j^1 and X_j^2 such that these are significantly differently distributed from each other as well as from the other groups in Π. Following the notion that latent factors are identifiable by distinct pattern distributions, we start the refinement process of a given group $X_j \in \Pi$ with a pattern $x \in S \times \mathcal{I}$ by separating a group into two children

$$X_j^x \equiv \{t \in X_j \mid x \subseteq t\} \text{ and } X_j^{\not x} \equiv X_j \setminus X_j^x \ .$$

The corresponding refinement of Π is written as

$$\text{refine}_\Pi(x, j) \equiv \{\Pi', A(\Pi')\} \ ,$$

where the new partitioning Π' is $\Pi \setminus \{X_j\} \cup \{X_j^x, X_j^{\not x}\}$.

As real data is noisy and distributions are complex, it is unlikely that an individual pattern x perfectly identifies a latent group. That is, after splitting a group, it may be that the overall assignment of transactions to groups is suboptimal with regard to the likelihood. Just like in the EM algorithm, we therefore iteratively reassign transactions to those groups where they achieve the highest likelihood. That is, in each iteration, we ensure for every $t \in X$ that

$$t \in X_{\hat{\imath}} \iff \hat{\imath} = \arg\max_{j \in [k]} p(t \mid S_j) \ , \tag{2.5.4}$$

re-estimate the distribution p, re-compute $A(\Pi, S)$, and repeat until convergence. Starting with $\Pi = \{X\}$, we iteratively refine the current partitioning by selecting the JS-significance refinement of Π with highest marginal gain, until convergence of ℓ. Formally, out of the set G of candidates

$$\left\{(j, x) \in [k] \times S_j \cup \mathcal{I} \;\middle|\; \text{refine}_\Pi(x, j) \text{ is significant}\right\} \ , \tag{2.5.5}$$

Algorithm 2.3: **DISC** for Discovering the Composition

Input: Data X, significance threshold α
Output: Partitioning Π, pattern set S, assignment A

$\Pi \leftarrow \{X\}$
$S, A \leftarrow \textsc{Desc}(X, \Pi)$
$G \leftarrow$ according Eq. (2.5.5)
while $G \neq \emptyset$ **and** ℓ has not converged
    $S, A \leftarrow \textsc{Desc}(X, \Pi)$
    $c \leftarrow \arg\min_{c \in G} \ell(\text{refine}_\Pi(c), S)$ *cf.* Eq. (2.5.6)
    $\Pi', A' \leftarrow \text{refine}_\Pi(c)$
    while ℓ has not converged
        let Π' satisfy Eq. (2.5.4)
        $A' \leftarrow A(\Pi')$ according to Eq. (2.5.3)
        let $S'_j = \{x_i \in S \mid A'_{ij} = 1\}$
        $p \leftarrow$ infer $p(\cdot \mid S'_j)$ for each group $X_j \in \Pi'$
    if $\ell(\Pi', S, A') < \ell(\Pi, S, A)$
        $\Pi \leftarrow \Pi'$, $A \leftarrow A'$
        $G \leftarrow$ according to Eq. (2.5.5)
    else
        $G \leftarrow G \setminus \{c\}$
return (Π, S, A)

we select the refinement candidate that reduces ℓ most

$$\arg\min_{(x,j) \in G} \ell(\text{refine}_\Pi(x, j), S) \ . \tag{2.5.6}$$

Putting all the above together, we have the DISC algorithm, whose pseudocode we give as Algorithm 2.3. In a nutshell, starting from the trivial partitioning $\Pi = \{X\}$, we characterize groups with patterns using DESC, use these patterns to find the best refinement of Π, reassign the rows to optimize the likelihood, and only accept this refinement if it is significant. We repeat this until convergence. Disregarding the complexity of inferring our distribution, DISC scales linearly with the size of the candidate set G. In the worst case, this means $|\Omega| \times |X|$. However, since in practice, both S and Π tend to be small, DISC is feasible on real-world data.

2.6 Related Work

The vast majority of the literature has been devoted to either finding clusters of transactions or finding patterns that characterize a dataset. Surprisingly, there exists no technique to discover the *pattern composition* of the data.

Our problem relates to mixture modeling [46], where data is modeled as a mixture of several probability distributions. Mixture modeling, however, requires us to assume a probability distribution, whereas the true distribution is unknown. Similarly, clustering [117] is related, as it groups data points, but it relies on an assumed distance measure. Additionally, in contrast to our approach, many existing approaches are stochastic or require us to choose the number of groups up front, and none characterize the commonalities and differences between the groups in interpretable terms.

In this sense, co-clustering, also known as bi-clustering, is more closely aligned with our goal. In co-clustering, we simultaneously cluster rows and columns, and we can interpret the column clustering as an implicit characterization of row clusters. Moreover, there exist parameter-free methods like information co-clustering [48] and cross-associations [30]. However, these techniques only discover non-overlapping rectangles in the data that are exceptionally dense or sparse, rather than data groups with significantly different pattern distributions.

Boolean matrix factorization [125, 127, 135] is closely related to our problem and bi-clustering [132], in that it seeks to express the data in terms of a pre-determined number of patterns. Although we can use the available information to partition the data [42], this does not necessarily result in significantly differently distributed groups. It is also not clear how to choose the right number k of to-be-identified patterns, although there exist strategies which involve various model selection criteria [7, 8, 11, 67, 81, 82, 195]. In practice, the right choice often results in a coarse-grained representation of the data that lacks details. Our method, on the other hand, not only identifies the right

number of patterns and significantly differently distributed groups, it also provides the necessary details for groups.

There also exist methods that can provide (post-hoc) explanations, for example using a consistent set of decision rules for predicting a single class [97], multiple classes [34, 147, 148], or a clustering [32, 89]. These rules together characterize the decision boundary for a cluster, whereas we are interested in those patterns that characterize the similarities and differences between groups. In other words, rather than explaining the clustering after the fact, our models *directly discover and justify* the clustering.

By nature, pattern mining methods are strongly related to Disc. Frequent pattern mining [5, 133] is well-known to discover far too many patterns for the result to be interpretable. opus [187, 189] curbs the pattern explosion through the use of statistical tests. Slim [164], iim [59], and mtv [118] are examples of techniques that discover concise and non-redundant pattern sets. These methods all only provide a single pattern set for a single database. DiffNorm [27] is the only method we know that, for a given data partitioning, can characterize differences and similarities between the pattern distributions.

2.7 Experiments

In our experiments, we evaluate Desc and Disc on synthetic data as well as on 17 real-world datasets that together span a wide variety of domains, sizes, and dimensionalities. All datasets we use are publicly available. We took *BMS VW, Adult, Page Blocks, DNA Ampl., Letter Recog. Anneal, MCADD, Led 7, Mammals, ICDM Abstracts, Waveform, Plants* from the UCI Machine Learning Repository and *Chess, Mushroom, Pumsb** from the Mining Dataset Repository.[1] The *DQ* dataset of lemmatized Deep-Learning and Quantum-Theory arXiv abstracts can be found in our online material. In Table 2.1, we provide basic information about the datasets and the minimum support used in our experiments. We implement Disc in C++, run experiments on a 12-Core Intel Xeon E5-2643 CPU, and report wall-clock time.

[1] https://archive.ics.uci.edu/ml, http://fimi.ua.ac.be/data/

In many of the following experiments, we compare the likelihood ℓ of the estimated model with the likelihood $\ell^{\emptyset}$ of the initial model, that is, $S^{\emptyset} = \mathcal{I}$, for a single group $\Pi = \{X\}$. We measure the likelihood ratio $\ell/\ell^{\emptyset}$, in percent, where a lower value corresponds to a higher regularized likelihood of the data under the model. In all experiments, we use the same significance level $\alpha = 0.01$.

Table 2.1: The sizes, dimensionality, number of classes, and the minimum support of patterns for datasets used in our experiments.

Dataset	$\lvert X\rvert$	$\dim X$	k	Min Support
BMS WV 1	59,602	497	1	32
Mushroom	8,124	23	2	10
Adult	48,842	15	2	5
Page Blocks	5,473	11	5	1
DNA Amp.	4,590	392	1	5
Chess Big	3,196	37	18	319
Let. Recog.	20,000	17	26	1
DQ	9,993	433	1	99
Anneal	898	71	5	1
Pumsb Star	49,046	7,116	1	12,500
MCADD	31,924	198	2	50
Chess	28,056	7	2	5
Led 7	3,200	8	10	40
Mammals	2,183	121	1	5
ICDM	859	3,932	1	10
Waveform	5,000	22	3	5
Plants	34,781	68	1	5

2.7.1 Describing Groups

First, we study our pattern-set miner Desc on real-world datasets. Before we characterize datasets for a given partitioning, we start with the special case of discovering a pattern set for a given composition. In this setup, we compare with mtv [118] and DiffNorm [27]. For efficiency reasons, these methods only consider frequent patterns, i.e., patterns for which $q(z) > \kappa$ according to a user-defined minimum frequency threshold κ (minimum support). It is trivial to constrain Desc to consider frequent patters only, and to compare fairly, we use the same thresholds for all methods in the following experiments.

We consider 9 labeled datasets, which we partition based on their class labels. DIFFNORM characterizes a pre-partitioned dataset Π, whereas MTV does not make use of any partitioning, i.e., it can only be applied to a complete dataset. As DESC can do both, we apply it to the partitioned data where a partition is available, and otherwise to the complete data.

In Figure 2.1a, we show that MTV and DESC discover pattern sets in the order of tens of patterns, while the pattern sets returned by DIFFNORM are often one order of magnitude larger. In Figure 2.1b, we report the wall-clock runtime of all methods. We see that DESC requires an order of magnitude less time than MTV to achieve its results. On average, DESC requires just 13% of the runtime of DIFFNORM, and only 0.24% of the runtime of MTV. As MTV and DESC optimize the same score, we can fairly compare their results. In Figure 2.1c, we show that DESC outperforms MTV in almost all cases when measured by the likelihood ratio of its results.

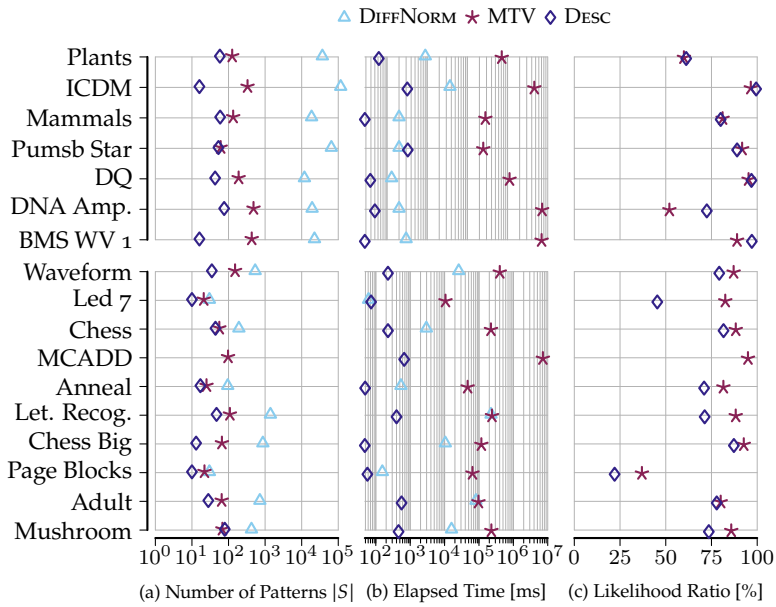


FIGURE 2.1: DESC efficiently discovers concise pattern sets. We show the number of discovered patterns (log scale) in Figure 2.1a, the runtime (seconds, log scale) in Figure 2.1b, and the likelihood ratio $\ell/\ell_\emptyset$ (lower is better) in Figure 2.1c. Each subfigure consists of two panels for unlabeled data (top panel) and labeled data (bottom panel).

2.7.2 Discovering the Composition

Now, we study the full algorithm: Simultaneously discover both the pattern sets and the partitioning of the dataset using Disc. First, we test and verify Disc on synthetic data with 128 items in $\mathcal{I}$. For this, we generate synthetic datasets such that we have access to the ground truth. In each trial, we randomly sample a dataset X, containing $1, 2, 4, 8$ groups. For each group $X_j \in \Pi$, we randomly generate and insert 5 characteristic patterns into S_j. For any disjoint pair X_i, X_j, we generate 3 shared patterns with probability of 20% that are inserted into both S_i and S_j. Every pattern has a randomly chosen frequency associated with it. Each group X_j consists of 256 rows. In each row, we insert each pattern from S_j with its corresponding frequency uniformly at random. Lastly, we introduce additive noise, by randomly inserting items into each row independently with probability of 5%. For each k^*, we sample 20 datasets and compare the ground-truth with Disc and Desc. On average, Disc reaches a likelihood ℓ^{Disc} within 2% of the ground truth, i.e., $\ell^{\text{Desc}_{\Pi^*}} \pm 2\%$, and always recovers the ground-truth number of groups k^*.

Having verified that Disc works on synthetic examples, we now study Disc on the real-world datasets. To do so, we measure how similar data within a group is, using ℓ, and how differently distributed the groups are, using the *pairwise symmetric KL-divergence* (PSKL)

$$\frac{1}{2\binom{k}{2}} \sum_{ij \in \binom{[k]}{2}} D\left(p_i \| p_j\right) + D\left(p_j \| p_i\right) \ ,$$

which averages the divergence between pairs of distributions.

Next, we compare Disc to clustering. While options include k-means, and Expectation-Maximization, these are not a good fit for our case as they are stochastic, require a number of clusters, and neither is the concept of a centroid well-defined on discrete spaces, nor is it clear what efficiently queryable distribution to use. In contrast, Dbscan [53] relies on a distance measure, which we can appropriately define as

$$d(s,t) = 1 - |s \cap t| \,/ \max(|s|, |t|) \,.$$

Our approach is as follows: First, we cluster the dataset using Dbscan and get $\Pi \in \varpi(X)$. Next, we use Desc to describe the clusters Π post-hoc by means of S, p, and $A(\Pi, S)$. Since Dbscan relies on a hyperparameter, we optimize ℓ using a grid search over 7 ϵ-candidates, and we do not constrain cluster sizes. We call this algorithm Dbdesc$_d$. Similarly, we define Dbdesc$_{d'}$, which uses a different distance measure

$$d'(s,t) = d(c(s), c(t)) \ ,$$

where $c(t)$ contains patterns from S that are subsets of t, i.e.,

$$c(t) = \{s \in S \mid s \subseteq t\} \ .$$

We apply Disc, Dbdesc$_d$, and Dbdesc$_{d'}$ to all 17 datasets without using any class labels, and summarize results in Figure 2.2. Note, for example, the much better likelihood ratio of Disc's composition in comparison to Dbdesc$_d$ and Dbdesc$_{d'}$. Overall, we see that Disc discovers diverging groups that have higher likelihood in comparison to the cluster-based composition from Dbscan.

We further compare Disc with DiffNorm, Desc on the full data, and Desc$_\Pi$ given the class-label decomposition on 9 labeled datasets, showcasing the results in Table 2.2. Almost always, we observe a significantly lower PSKL-divergence between classes than between groups. Additionally, Disc's model usually has a significantly lower objective function ℓ than the class-based result from Desc. This suggests that classes are not always a good indicator for groups. Furthermore, we see a higher likelihood for decomposed data in comparison to Desc without decomposing the data on class labels, shown in Table 2.3.

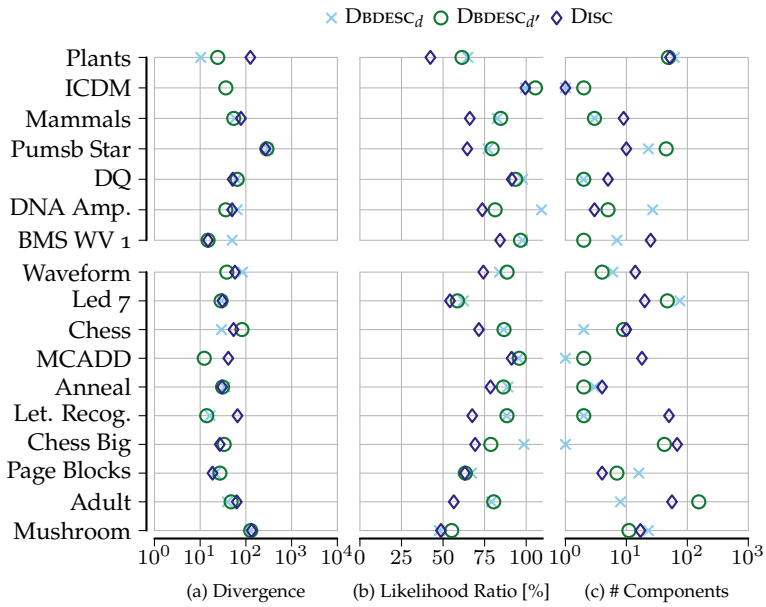


FIGURE 2.2: DISC discovers informative and interpretable compositions. We show the PSKL (higher is better) in (a), the likelihood ratio $\ell/\ell_\emptyset$ (log scale, lower is better) in (b), and the number of discovered groups (log scale) in (c). Each subfigure consists of two panels for unlabeled data (top panel) and labeled data (bottom panel).

Table 2.2: Disc discovers informative patterns. For 9 datasets with class labels, we compare DiffNorm, Desc on the full data, Desc_Π given the class-label decomposition, and Disc. We give the true number of classes (k), the number of groups that Disc discovers ($\hat{k}$), the number of patterns discovered ($|S|$), the likelihood ratio ($\ell/\ell_\emptyset$, lower is better), the class-label purity of the groups discovered by Disc (higher is better), the divergence (PSKL, higher is better) between the ground truth and the discovered groups, and the runtime (s, lower is better).

			$\lvert S\rvert$				$\ell/\ell_\emptyset$ [%]			Purity	PSKL		Time	
Dataset	k	$\hat{k}$	DiffNorm	Desc	Desc_Π	Disc	Desc	Desc_Π	Disc	Disc	Desc_Π	Disc	Desc	Disc
Adult	2	56	719	28	28	29	88.9	83.1	56.5	0.8	8.3	63.3	899ms	11h36m
Anneal	5	4	95	17	17	17	91	83.9	78.5	0.8	31.1	30.1	101ms	0m17s
Chess	2	10	192	44	44	44	88.1	84.7	71.6	0.6	5.6	53.7	278ms	18m39s
Chess Big	18	68	869	13	13	13	98	91.4	69.3	0.3	9.2	26.9	55ms	52m19s
Led 7	10	20	30	10	10	10	89.6	65.5	54.2	0.7	17	30.6	45ms	1m54s
Let. Recog.	26	50	1398	47	47	49	88.9	79	67.6	0.3	20.5	65.5	710ms	15h52m
Mushroom	2	17	424	66	79	70	82	77	48.7	1	59.6	133.6	660ms	32m24s
Page Blocks	5	4	30	10	10	10	98.2	68	63.2	0.9	4.9	18.6	95ms	0m14s
Waveform	3	14	535	35	35	36	89.1	82.2	74.2	0.7	31.6	57.7	309ms	1h45m

Table 2.3: For 8 unlabeled datasets without class labels, for Desc and Disc, we report the number of patterns ($|S|$) and runtime (s), and for Disc, we additionally report the number of groups discovered ($\hat{k}$), the gain in likelihood ($\ell/\ell_\emptyset$) in percent, and the PSKL-divergence between the discovered groups.

	Desc		Disc				
Dataset	$\lvert S\rvert$	$\ell/\ell_\emptyset$	$\hat{k}$	$\lvert S\rvert$	$\ell/\ell_\emptyset$	PSKL	Time
BMS WV 1	16	97.1	25	28	84.3	14.8	3h21m
DNA Amp.	76	73.1	3	84	73.6	50	1m6s
DQ	43	96.9	5	55	91.3	51.4	11m45s
ICDM	16	99.5	1	16	99.5	0	2m0s
MCADD	95	94	18	104	91.2	41.2	1h13m
Mammals	59	80.4	9	64	66.1	78.1	7m34s
Plants	58	61.2	52	60	42.4	125.3	13h5m
Pumsb Star	53	84.9	10	53	64.6	268.4	1h34m

2.7.3 Qualitative Study

Finally, we study the interpretability of the composition discovered by Disc and evaluate it qualitatively by manually inspecting the pattern composition of two datasets.

European Mammals

First, we consider the *Mammals* dataset provided by the European Mammal Society. This dataset consists of presence records of 124 European mammals within areas of 50-by-50 kilometers. Additional geographical information was not used during the experiments.

Disc discovers 9 groups with 64 patterns in total. We geographically depict the groups it identifies in Figure 2.3. Although Disc does not know the spatial locations of the data points, it discovers (almost completely) contiguous areas in Europe that correspond to ground-truth habitats. Moreover, the patterns it discovers for these groups are meaningful: For example, although the combination of species as *Wolverine* and *Norway Lemming* are highly characteristic for both "Scandinavian" groups, their distribution differs between these groups. For the Iberian Peninsula, the *Common Genet* and *Mediterranean Pine Vole* are discovered to be very characteristic. The habitation zone of the

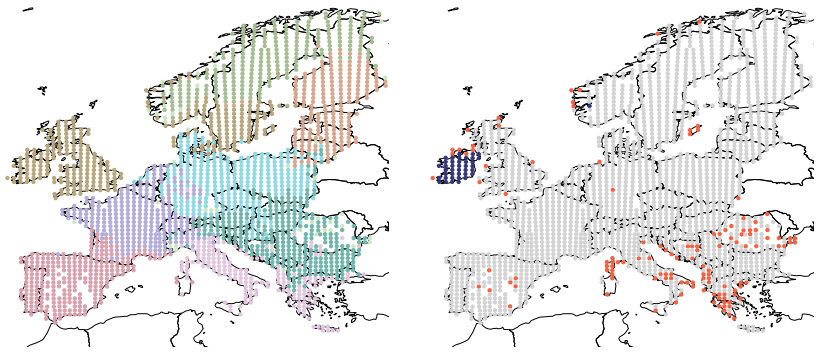


Figure 2.3: Disc discovers meaningful partitions. We show results of Disc (left) on the *Mammals* dataset. The 9 groups represent contiguous areas that correspond to known habitats. Dbscan (right) essentially only discovers Ireland.

latter spreads to Southern France, and this is reflected by this pattern being shared between these two groups. Further, Disc discovers that the co-occurrence of *Eurasian beaver, Red squirrel* is descriptive across Europe. Last but not least, Disc finds that the *Eurasian Harvest Mouse, European Mole, Eurasian Water and Pygmy Shrew, Stoat, Field Vole* are all very common across Europe, and includes them in a single pattern shared among most groups spanning Europe.

Topic Analysis

Next, we study the composition of the *DQ* dataset. This is a dataset that consists of 10 000 abstracts crawled from arXiv. Half of the abstracts is from papers on Deep Learning, the other half is from papers on Quantum Theory.

From these abstracts, we remove stop words, extract and lemmatize nouns and verbs, erase words with a frequency lower than 0.05, and remove the class labels. Overall, we have a dataset over 433 items. The composition that Disc discovers on this data consists of 5 groups and 224 patterns. The "Deep Learning" class is covered by 3 groups, spanning 36%, 53%, and 10% of the papers on deep learning, respectively. The "Quantum" class consist of 2 groups which cover 39.9%

TABLE 2.4: DISC discovers interpretable compositions. We show a selection from the pattern composition discovered on the *Deep Quantum* dataset. Overall, DISC discovers 223 characteristic and common patterns as well as 5 groups, two of which consist of mostly "Quantum Physics" papers, and three of which of mostly of "Deep Learning" papers.

	QUANTUM PHYSICS
$S_{1,2}$	*local entanglement, Bell inequality, standard model*
S_1	*standard approach, learn data, research paper*
S_2	*probability distribution, computer computation, search algorithm*
	DEEP LEARNING
$S_{3,4,5}$	*neural networks, hidden layer, computer vision*
	gradient descent, adversarial attack
$S_{3,4}$	*information prediction, space representation*
S_3	*neural processing, reinforcement environment agent*
S_4	*feature representation, learning challenge, training optimization*
	COMMONALITIES
$S_{1–5}$	*experimental results, lower bound, Hilbert space*

and 59.9% of the quantum papers. Overall, the groups have a total purity of more than 99.5%. In Table 2.4, we give examples of patterns discovered by DISC. Common patterns across all papers include *experimental results*, *lower bound*, and *Hilbert space*, whereas a pattern such as *reinforcement agent environment* is only characteristic for one group of the deep learning papers.

2.8 CONCLUSION

We studied the novel problem of discovering the composition, that is, a partitioning of the dataset and its description using locally characteristic patterns and patterns shared across sets of groups. We formalized this problem in terms of the family of maximum entropy distributions over itemsets, and defined the best composition as the one that gives the most succinct description of the data.

We introduced a highly efficient pattern miner for single and multiple datasets, DESC, to succinctly describe one or more data groups,

beating the state of the art in descriptiveness, conciseness and runtime. We also demonstrated how Desc can be used to describe the result of a clustering algorithm. This allowed us to observe that jointly optimizing for interpretability and likelihood is doable in practice, and that it can outperform clustering algorithms like Dbscan with a post-hoc explanation.

Building on Desc, we developed Disc to discover the pattern composition of a dataset. Experimental evaluation showed that Disc efficiently discovers interesting, meaningful, and easily interpretable pattern compositions from data. The data groups we identify are described concisely, by characteristic and shared patterns. Disc explains why there are groups in binary tabular data, what makes them special, and what is common among them—via insightful patterns.

3
Differentially Describing Groups of Graphs

In the previous chapter, we sought insightful patterns in *binary tabular data* with unknown groups, and we found such patterns in the pattern composition of those data. Shifting gears toward a more complicated, yet equally relevant data type, in this chapter, we will seek insightful patterns in *groups of graphs*. In particular, this chapter is motivated by questions like: How does neural connectivity in autistic children differ from neural connectivity in healthy children or autistic youths? What patterns in global trade networks are shared across classes of goods, and how do these patterns change over time? To answer such questions, we seek to differentially describe groups of graphs: Given a set of graphs and a partition of these graphs into groups, discover what graphs in one group have in common, how they systematically differ from graphs in other groups, and how multiple groups of graphs are related. We refer to this task as *graph group analysis*, which seeks to describe similarities and differences between graph groups by means of statistically significant subgraphs. To perform graph group analysis, we introduce Gragra, which uses maximum entropy modeling to identify a non-redundant set of subgraphs with statistically significant associations to one or more graph groups. Through an extensive set of experiments on a wide range of synthetic

This chapter is based on the publication: Coupette, Dalleiger, and Vreeken [35].

and real-world graph groups, we confirm that Gragra works well in practice.

3.1 Introduction

Differentially describing groups of graphs lies at the heart of many scientific and societal challenges. Neuroscientists, for example, want to characterize brain activity in healthy subjects, elucidate how it differs from brain activity in subjects diagnosed with certain disorders or diseases (e.g., autism or Alzheimer's), and investigate whether their findings are the same across different groups of subjects (e.g., children, adolescents, and adults; or men and women). Policymakers, security experts, and epidemiologists alike could seek to understand patterns of human mobility, be it to improve the resilience of traffic infrastructure to random failures and targeted attacks, or to curb the spread of infectious diseases. And international economists might want to investigate patterns of world trade, e.g., imports and exports between countries, and ask how these vary across different years and product classes.

We refer to the common task underlying these scenarios as *graph group analysis*: Given a set of graphs and a partition of this set into *graph groups*, succinctly summarize the commonalities and differences between graphs in the same group, between graphs in different groups, and between the relationships connecting the groups. In this chapter, we formalize graph group analysis as a maximum likelihood modeling problem, using *significant subgraphs* as graph patterns to factorize our probability distribution. We introduce Gragra (*Gra*ph *gr*oup *a*nalysis) as an algorithm to solve this problem, which jointly discovers a set of graph patterns and an assignment of these patterns to graph groups.

As a real-world example of graph group analysis, consider Figure 3.1. Here, we show the top shared (left) and specific (right) patterns identified in resting-state functional brain networks of adolescents with and without autism spectrum disorder, where nodes in the graphs correspond to brain regions, and edges signal strong connectivity between regions. On the right, patterns with red edges are characteristic

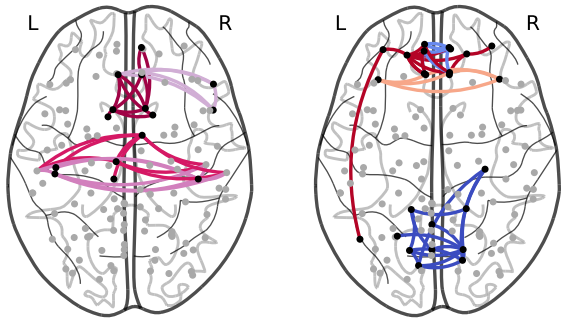


Figure 3.1: Gragra discovers common and contrastive graph patterns in noisy, heterogeneous groups of graphs, capturing, e.g., systematic similarities (left) and differences (right) between the functional brain networks of adolescents with and without autism spectrum disorder. Here, nodes represent centers of mass for brain regions from the AAL Atlas, and edge color classes correspond to significant subgraphs shared between (left) or specific to (right) groups, with individual edges signaling strong connectivity between regions.

of autistic adolescents, and patterns with blue edges are characteristic of non-autistic adolescents. They indicate over- and underconnectivity, respectively, in the brains of autistic adolescents when compared to typically developed controls. Although there is no consensus regarding the relationships between autism and neural connectivity [79], our method identifies graph patterns that permit neuroscientific interpretation: For example, the dark blue pattern in Figure 3.1 indicates underconnectivity between the visual cortex, responsible for processing visual information, and the lingual gyrus, involved in vision and word processing.

Graph group analysis is related to *graph classification* [e.g., 101], but we are interested not only in what is *different* but also in what is *similar* among our graph groups. Our task further shares some of its motivation with *significant subgraph mining* [e.g., 168], *graph summarization* [e.g., 105], and *data clustering* with graphs as data points [e.g., 130]. However, we focus on a *complete* characterization of a *set* of graphs under a *given* partition—a cornerstone of scientific discovery involving graph data.

The remainder of this chapter is structured as follows. After settling our basic notation in Section 3.2, we describe the theoretical

foundations of our method in Section 3.3 and introduce our algorithm in Section 3.4. Having covered related work in Section 3.5, through experiments on synthetic and real-world data, we demonstrate that Gragra works well in practice in Section 3.6, before rounding up with discussion and conclusions in Section 3.7.

3.2 Preliminaries

We consider a set $\mathcal{G} = \{G_1, \ldots, G_{|\mathcal{G}|}\}$ of $|\mathcal{G}|$ *node-aligned* graphs $G_i = (V, E_i)$ with $n = |V|$ nodes and $m_i = |E_i|$ edges, omitting the subscripts when clear from context. A partition $\Pi = \{\mathcal{G}_1, \ldots, \mathcal{G}_k\}$ is a set of k non-empty subsets of $\mathcal{G}_i \subseteq \mathcal{G}$, called *graph groups*, of cardinalities $c_i = |\mathcal{G}_i|$, whose disjoint union is $\mathcal{G}$. Our graphs can be undirected or directed, loopy or non-loopy, and unweighted, edge-labeled, or integer weighted, where for the purposes of our model, we treat distinct edge labels or edge weights as a set W of categories, and edges $e \in E_i$ are drawn from the set $\mathcal{E} = V \times V \times W$ of all possible weighted edges.

The *empirical frequency* of edge set $x \subseteq \mathcal{E}$ in group $\mathcal{G}_i$ is

$$q_i(x) = |\{(V, E) \in \mathcal{G}_i \mid x \subseteq E\}| \,/ c_i \ ,$$

and we denote by V_x the set of nodes incident with at least one edge in x.

Observing that our edge sets contain discrete variables, we reuse the maximum entropy distribution introduced in Section 2.3, thus benefitting from an unbiased and information theoretically sound probabilistic modelling. That is, we model the *expected frequency* of x in $\mathcal{G}_i$ under a given set of edge sets $S \subseteq 2^{\mathcal{E}}$ as $p_i(x \mid S) = \mathbb{E}_f[x \mid S]$ for the maximum entropy distribution f corresponding to $\mathcal{G}_i$.

3.3 Theory

We now lay the theoretical foundations of our method, introducing our probabilistic model, our objective function, and our statistical test. At a high level, our goal in graph group analysis is to discover a set S of *graph patterns*, i.e., edge sets of *connected subgraphs*, and an *association*

matrix A assigning graph patterns to graph groups, such that S and A together reveal the similarities and differences between graphs in the same group, between graphs in different groups, and between the relationships connecting the groups. A pattern is *specific* if we assign it to only one graph group, and it is *shared* if we assign it to several graph groups. We choose which patterns to include in our model based on the information we gain from them, testing whether this gain is statistically significant to rule out spurious results.

To avoid redundancy, we assign a pattern x to a group $\mathcal{G}_i$ iff x is *informative* for $\mathcal{G}_i$, given what we already know about all groups. More precisely, using x as a column index of A in a slight abuse of notation, we set $A_{ix} = 1$ iff x is *informative* for $\mathcal{G}_i$ under our current model (S, A). We assess this by comparing the *empirical frequency* of x in group $\mathcal{G}_i$, $q_i(x)$, to its *expected frequency* in that group under our current model, $p_i(x \mid S_i)$, where $S_i = \{x \in S \mid A_{ix} = 1\}$, and p_i is obtained from a practical approximation of the maximum entropy distribution with constraint set S_i. x is *informative* for $\mathcal{G}_i$ iff $q_i(x)$ is significantly different from $p_i(x \mid S_i)$, as judged by a statistical test, and we add x to S (and column x to A) if x is informative for *some* $\mathcal{G}_i \in \Pi$.

To identify a suitable set of graph patterns S and an adequate association matrix A, we exploit the interplay between two steps. First, we discover the best pattern x to add to S, given the current (S, A), and second, we identify the best assignment of x to graph groups to update A, given the current (S, A) *and* a new pattern x. We now describe each step in more detail.

3.3.1 Identifying Informative Graph Patterns

To measure the likelihood of a set $S \subseteq 2^{\mathcal{E}}$ of graph patterns, we use the *Bayesian Information Criterion* (BIC),

$$\mathrm{BIC}(S) = \ell(S) + (k \cdot |S|)/2 \log |\mathcal{G}| \quad [158],$$

where $k \cdot |S|$ is the number of coefficients in our model, and

$$\ell(S) = \sum_i \ell_i(S) = -\sum_i \sum_{G \in \mathcal{G}_i} \log p_i(G \mid S_i) \,,$$

is the log likelihood of S (with $S_i \subseteq S$ derived from A), assuming that the graphs in a group are independent and identically distributed. This allows us to identify a good set of graph patterns by minimizing the BIC score, i.e.,

$$\arg\min_{S \subseteq 2^{\mathcal{E}}} \{\mathrm{BIC}(S)\} \,.$$

Solving this problem exactly poses significant challenges in practice due to its combinatorial nature and the explosion in the number of solution candidates. Therefore, we employ a greedy search strategy, iteratively selecting the graph pattern $x \subseteq \mathcal{E}$ that best improves our current model. That is, for a given (S, A), we select the graph pattern x that maximizes our likelihood, or equivalently, maximizes the difference

$$\mathrm{BIC}(S) - \mathrm{BIC}(S \cup \{x\}) \,,$$

which we write as

$$\Delta(x) = \ell(S) - \ell(S \cup \{x\}) - k/2 \log |\mathcal{G}| \,.$$

In a nutshell, the core of our approach is the procedure

$$S \leftarrow S \cup \left\{ \arg\max_{x \subseteq \mathcal{E},\ \Delta(x) > 0} \{\Delta(x)\} \right\} , \tag{3.3.1}$$

by which we iteratively and greedily insert into S the pattern $x \subseteq \mathcal{E}$ that locally maximizes our information gain.

If we only have a limited number of samples, we cannot tell if our information gain is due to random fluctuations or due to signal, using a model selection criterion alone. To ensure that we have significant patterns, we add x to S only if its information gain $\Delta(x)$ is *statistically*

significant. Therefore, we test whether we can reject the null hypothesis

$$H_0 : \ \mathrm{BIC}(S) = \mathrm{BIC}(S \cup \{x\}) \ .$$

To this end, we use Vuong's closeness test [182], a likelihood ratio test designed for model selection problems under BIC. Vuong's test statistic is defined as $2\Delta(x)$, which is asymptotically χ^2-distributed with $\mathrm{df}_\Delta(x) = \mathrm{df}\, p_i(\,\cdot \mid S \cup \{x\}) - \mathrm{df}\, p_i(\,\cdot \mid S)$ degrees of freedom. To calculate $\mathrm{df}_\Delta(x)$, we count the coefficients θ that must be changed in every distribution if we insert x into S. As we add one coefficient for x, and update at least $|x|$ edge coefficients per group, we arrive at $|x| + 1$ additional degrees of freedom.

3.3.2 Discovering Differential Pattern Associations

Once we have selected a new pattern $x \subseteq \mathcal{E}$ to add to S, given the current S and A, we identify a good assignment of x to graph groups $\mathcal{G}_i \in \Pi$ to update A. Here, the significance of $\Delta(x)$, which is used to accept x into S, signals that x is informative for *some* $\mathcal{G}_i \in \Pi$, but it does not tell us for *which* $\mathcal{G}_i$. To assign x to a group $\mathcal{G}_i$, we hence rely on the *partial* information gain of x for $\mathcal{G}_i$,

$$\Delta_i(x) = \ell_i(S_i) - \ell_i(S_i \cup \{x\}) - k/2 \log |\mathcal{G}| \ .$$

Again, we use Vuong's closeness test to decide whether $\Delta_i(x)$ is significant; and if it is, we set $A_{ix} = 1$.

3.4 Algorithm

Having established its theoretical groundwork, we now introduce Gragra as an algorithm to differentially describe groups of graphs using sets of significant subgraphs. Gragra, whose pseudocode is given as Algorithm 3.1, revolves around the procedure stated in Eq. (3.3.1), a greedy process that iteratively selects the graph pattern candidate that best enhances our model. Hence, rather than exhaustively searching for the best graph patterns, we propose to grow graph patterns by systematically adding edges to candidates.

To enable our model to infer all possible graphs, we initialize it with the set $\mathcal{E}$ of all possible edges. As our initial graph to grow, we then select the most promising graph pattern from our initial candidates, i.e., the connected triples

$$C = \{\{x, y\} \mid x, y \in \mathcal{E}, x \neq y, V_{\{x\}} \cap V_{\{y\}} \neq \emptyset\} \ .$$

Starting with a graph pattern
X, we explore all its expansions,

$$((V_x \times V \times W) \cup (V \times V_x \times W)) \setminus x \ ,$$

from which we select the best candidate pattern to grow further, as long as we gain information and $\Delta(x)$ is significant. We summarize these steps in the function Grow of Algorithm 3.1 (l. 13–21).

Grow requires many inferences of Δ, which involve inferring many more expected frequencies p_i, rendering exact computation impractical. We thus design a practical, pessimistic heuristic that only considers the information gain from graphs $G \in \mathcal{G}$ in which x is fully present: Starting with $\Delta(x)$, and abbreviating the constant model cost delta

$$k/2 \cdot |S \cup \{x\}| \cdot \log |\mathcal{G}| - k/2 \cdot |S| \cdot \log |\mathcal{G}| = k/2 \log |\mathcal{G}|$$

as c, we obtain

$$\begin{aligned}
\Delta(x) &= \ell(S) - \ell(S \cup \{x\}) - c \\
&= -\sum_i \sum_{G \in \mathcal{G}_i} \log p_i(G \mid S) - \log p_i(G \mid S \cup \{x\}) - c \\
&= -\sum_i \sum_{G \in \mathcal{G}_i} \log \frac{p_i(G \mid S)}{p_i(G \mid S \cup \{x\})} - c \ .
\end{aligned}$$

Now, by constraining the sum to include *only* graphs in which x is *fully* present, we get

$$-\sum_i \sum_{G \in \mathcal{G}_i, x \subseteq G} \log \frac{p_i(x \mid S)}{p_i(x \mid S \cup \{x\})} \frac{p_i(G \setminus \{x\} \mid S)}{p_i(G \setminus \{x\} \mid S \cup \{x\})} - c \ ,$$

using a factorization of p and G. By assuming that

$$\log \frac{p_i(G \setminus \{x\} \mid S)}{p_i(G \setminus \{x\} \mid S \cup \{x\})} \approx 0 \ ,$$

and since $p_i(x \mid S \cup \{x\}) = q_i(x)$ holds, we can further simplify the above to

$$-\sum_i c_i \cdot q_i(x) \log \frac{p_i(x \mid S)}{q_i(x)} - c = \sum_i c_i \cdot q_i(x) \log \frac{q_i(x)}{p_i(x \mid S)} - c \ ,$$

thus arriving at our heuristic

$$h(x) \equiv \sum_i h_i(x) = \sum_i c_i \cdot q_i(x) \log \frac{q_i(x)}{p_i(x)} - k/2 \log |\mathcal{G}| \ .$$

Subsequently, we use $h(x)$ instead of $\Delta(x)$ because it involves inferring only *one* expected frequency per graph group.

To summarize, Gragra proceeds as follows. Starting with an initial set of candidates C (l. 3), we select (l. 14) and grow (l. 15) the best candidate, and retain all significant expansions (l. 16), until we have grown x to its fullest potential (l. 18–19). Afterwards, we test if the information gain provided by x is significant, and if so, we keep track of its graph group associations (l. 8), and insert x into S (l. 9).

The computational complexity of Gragra depends on the number of candidates, which can grow to at most $|2^{\mathcal{E}}|$. In practice, Gragra's complexity depends on the number of times we grow graph patterns, which is data-dependent and bounded by the size γ of the largest connected component observed in an input graph, as growing beyond that reduces the information gain. Multiplying γ by the initial set of candidates, Gragra achieves a complexity of $\mathcal{O}\left(\binom{n}{3}|W|\gamma\right)$ for all practial purposes, where we assume that the complexity of inferring the expected frequency is bounded.

3.5 Related Work

To the best of our knowledge, we are the first to differentially describe groups of graphs through sets of significant subgraphs. Our method is inspired by advances in graph similarity description (Momo, [36]) and explainable pattern-set mining using maximum-entropy modeling as described in Chapter 2 (Disc, [43, 44]). However, Momo focuses on pairs and unpartitioned sets of graphs; DISC is designed for itemset data, ignores graph structure, and does not scale on graphs; and neither method uses a statistical test to select patterns. Further related

Algorithm 3.1: **Gragra**

Input: groups of graphs $\mathcal{G}_1, \ldots, \mathcal{G}_k$
Output: set of graph patterns S, association matrix A

$S \leftarrow \mathcal{E}$
$A \leftarrow$ empty binary matrix with k rows and 0 columns
$C \leftarrow \{\{x, y\} \mid x, y \in \mathcal{E}, x \neq y, V_{\{x\}} \cap V_{\{y\}} \neq \emptyset\}$
while $C \neq \emptyset$
    $\hat{x}, C \leftarrow$ Grow(C)
    if $\exists i \in [k]$ *s.t.* $h_i(\hat{x})$ is significant
        resize A
        $A_{i\hat{x}} = 1 \iff h_i(\hat{x})$ is significant $\forall i \in [k]$
        $S \leftarrow S \cup \{\hat{x}\}$
        estimate $p_i(\cdot \mid S_i)$ $\forall i \in [k]$ s.t. $A_{i\hat{x}} = 1$
return $S \setminus \mathcal{E}, A$

Fn. *Grow*(C)
    $x \leftarrow \arg\max_{x \in C} \{h(x)$ s.t. $h(x)$ is significant$\}$
    $C \leftarrow C \cup (((V_x \times V \times W) \cup (V \times V_x \times W)) \setminus x)$
    $C \leftarrow \{x \in C \mid h(x)$ is significant $\}$
    $\hat{x} \leftarrow \arg\max_{x \in C} \{h(x)\}$
    if $h(\hat{x}) > h(x)$
        return Grow(C)
    else
        return $\hat{x}, C \setminus \{\hat{x}\}$

work broadly falls into two categories: statistical inference on network populations, and graph mining for groups of graphs.

Statistical Inference on Network Populations. In the statistics literature, the task of analyzing multiple graphs simultaneously is typically framed as an inference problem for network-valued random variables [51, 111, 114]. Here, Ghoshdastidar et al. [62] establish limits for distinguishing two population distributions given small sample sizes, and Lunagómez, Olhede, and Wolfe [114] propose notions of mean and dispersion for a single population of networks, where the population mean is itself a network. Maugis et al. [122] use subgraph counts to test if all graphs in a sample are drawn from the same distribution, and Signorelli and Wit [161] propose a model-based clustering approach to describe subpopulations within a population of networks. Finally, Durante, Dunson, and Vogelstein [51] extend latent space approaches designed for single graphs to capture the probabilistic mechanism that generates multiple graphs from a single population distribution. Their model has been used to characterize and test for differences between groups of brain networks [52]—an actively studied application for which numerous statistical methods, mostly focusing on *testing* for differences, have been developed [63, 96, 102, 112, 113, 184].

Prior work in the statistics literature has centered on describing *one* network population or distinguishing *two* populations. In contrast, with Gragra, we aim to construct a *differential description* of *any* number of populations. Furthermore, we ask not only *if* these populations are different, but also *how* they are different and how they are *similar*.

Graph Mining for Groups of Graphs. In the graph mining literature, groups of graphs have been studied in contexts as diverse as significant subgraph mining [107, 168], graph classification [98, 180, 197], graph clustering with graphs as data points [130], anomalous graph detection [65], and graph summarization for time series of graphs [159]. Significant subgraph mining commonly considers small, node-labeled graphs with unaligned node sets, and hence, does not target our problem. However, our setup (medium-sized graphs with aligned node sets) has received heightened attention in the graph classification

community, again inspired by challenges from neuroscience [98, 180, 197].

The methods that are closest to our work are *contrast subgraphs* [98] and *signal subgraphs* [180], both designed for two groups of node-aligned graphs. *Contrast subgraphs* discover the densest subgraph in the difference of the summary graphs of the input groups (obtained by adding the graphs in each group separately and then subtracting the results), where the size of this subgraph depends on a user-specified regularization parameter α. *Signal subgraphs* assume edge independence as a prior to rank edges by the p-values of an edge-wise statistical test for distributional difference (e.g., Fisher's exact test). Like *signal subgraphs*, Gragra combines ideas from structural and statistical pattern mining to produce interpretable results that—unlike *contrast subgraphs*—are based on a statistical foundation. Gragra is more exploratory and more flexible than both competitors, however, because it treats graph group description as an end in itself and can handle any number of graph groups.

3.6 Experiments

We now present an extensive evaluation of our algorithm. To this end, we implement Gragra in C++ and expose a Python interface to facilitate experimentation. We run our experiments on Intel E5-2643 CPUs with 128 or 256 GB RAM, testing at a conservative significance level of 1×10^{-7} (or 1×10^{-5} when operating with less than 50 samples). Our experiments revolve around two questions:

1. Can Gragra reliably recover the ground truth from groups of synthetic graphs?
2. Does Gragra discover meaningful patterns in groups of real graphs?

3.6.1 Recovering Ground Truth from Synthetic Graphs

To assess the reliability of Gragra, we run it on groups of synthetic graphs with planted patterns. We consider three scenarios, namely,

1. summarizing *one* group of graphs,
2. differentially describing *two* groups of graphs, and

3. differentially describing *four* groups of graphs.

In all three scenarios, each graph group consists of 100 graphs with 100 nodes, and our configurations differ in their planted patterns (type, prevalence, and position) and noise levels.

For each configuration from Table 3.1, we generate 100 graph group datasets with $k \in \{1, 2, 4\}$ graph groups. Each group consists of 100 graphs with $n = 100$ nodes (labeled from 0 to 99), and edges are sampled randomly using a $G(n, p)$ random graph model, edge probability $p \in \{0.1, 0.2\}$, and different seeds. We then plant *cliques* (i.e., complete graphs) of size 5, *stars* (i.e., one hub node connected to pairwise nonadjacent spoke nodes) of size 10, and balanced *bicliques* (i.e., two equally sized independent node sets A and B such that every node in A is connected to every node in B) of size 10 as patterns into these random graphs, using the prevalence and position parameters given in the fourth and fifth columns of Table 3.1. Here, each column in the

TABLE 3.1: Synthetic graph group configurations. k is the number of groups, p is the edge probability in a $G(n, p)$ random graph model, P is the pattern (*cl*ique, *st*ar, or *bic*lique), and $|P|$ is the size of (the node equivalence classes in) the pattern. *Prevalence* is the occurrence probability of the pattern in the graph group, *position* is the label of the first node in the pattern, and t indicates the pattern *t*ype, i.e., whether it is *s*hared, *o*verlapping, or *c*ontrastive between graph groups.

k	p	$P(\|P\|)$	**Prevalence**	**Position**	t
1	0.2 0.1	$\begin{bmatrix}\text{cl}(5)\\ \text{st}(1,9)\\ \text{bc}(5,5)\end{bmatrix}$	$[0.2\ 0.2\ 0.2]^T$ $[0.1\ 0.2\ 0.3]^T$	$\begin{bmatrix}0\\5\\15\end{bmatrix}$	–
2	0.2	st(1, 9)	[0.2 0.2] [0.2 0.4] [0.4 0.4]	[0 0] [0 0] [0 10]	s c c
2	0.2	cl(5)	[0.2 0.2]	[0 0] [0 2] [0 5]	s o c
4	0.2 0.1	$\begin{bmatrix}\text{cl}(5)\\ \text{cl}(5)\\ \text{st}(1,9)\\ \text{st}(1,9)\\ \text{bc}(5,5)\end{bmatrix}$	$\begin{bmatrix}0.2&0.2&0&0&0\\0&0.2&0.2&0&0\\0&0&0.2&0.2&0\\0&0&0&0.2&0.2\end{bmatrix}^T$ $\begin{bmatrix}0.1&0.2&0&0&0\\0&0.1&0.2&0&0\\0&0&0.3&0.2&0\\0&0&0&0.3&0.2\end{bmatrix}^T$	$\begin{bmatrix}0\\5\\10\\20\\30\end{bmatrix} * 4$	$\begin{bmatrix}\text{c}\\ \text{s}\\ \text{s}\\ \text{s}\\ \text{c}\end{bmatrix}$

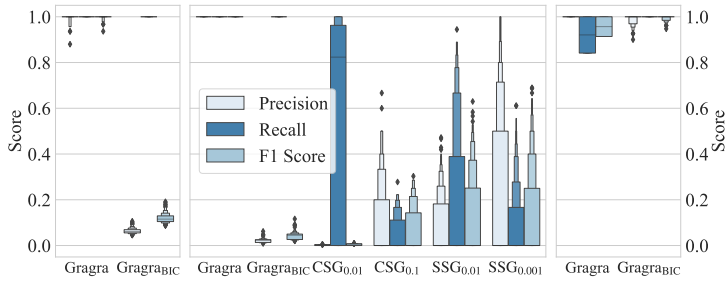


Figure 3.2: Gragra reliably recovers the ground truth from synthetic data. We show precision, recall, and F1-score distributions for Gragra, Gragra$_{BIC}$, *contrast subgraphs* (CSG), and *signal subgraphs* (SSG), separately for all experiments in our three different settings: one-group setting (left), two-group setting (middle), and four-group setting (right). Subscripts of CSG labels correspond to different choices of their regularization parameter α, and subscripts of SSG labels indicate different requirements for the p-values obtained from their edge-wise distributional difference test.

prevalence and position matrices corresponds to a graph group, and repeated columns in the four-group setting are condensed as $[\cdot] * 4$.

For example, for the second one-group setting (Table 3.1, Row 2), we plant a clique starting at node 0 with prevalence 0.1, a star starting at node 5 with prevalence 0.2, and a biclique starting at node 15 with prevalence 0.3, into 100 graphs generated using $G(100, 0.1)$.

For each scenario, we report the distribution of precision, recall, and F1 score, computed separately for each group of graphs, for the edges of the planted patterns across 100 graph group datasets sampled with different seeds. In all scenarios, we compare Gragra, which uses BIC with Vuong's closeness test for pattern selection, with a variant using only BIC and no statistical test to select patterns (Gragra$_{BIC}$). For configurations in the second scenario, we also compare our results with those from *contrast subgraphs* (CSG) and *signal subgraphs* (SSG), described in the previous section.

As we show in Figure 3.2, Gragra$_{BIC}$ delivers better results in the four-group scenario but generally has worse precision than Gragra, by treating noise as signal. CSG and SSG identify only constrastive patterns, and fail even for contrastive patterns if the individual edges in planted patterns have similar occurrence probabilities across groups.

Gragra, however, reliably recovers the ground truth across scenarios and configurations, which allows us to hope that it will also work in practice.

3.6.2 Discovering Meaningful Patterns in Real Graphs

To determine whether Gragra discovers meaningful patterns in groups of real graphs, we run 29 experiments on data of various graph types from three domains: functional brain networks (undirected, unweighted), air transportation networks (directed, weighted), and international trade networks (directed, weighted).

The functional brain network data stem from the Autism Brain Imaging Data Exchange (ABIDE). In the graphs representing these data, each node corresponds to a region of interest (ROI) from the automated anatomical labeling (AAL) atlas, and each unweighted, undirected edge corresponds to a relatively strong blood-oxygen-level dependent (BOLD) signal correlation between the time series of these regions obtained during a resting-state functional magnetic resonance imaging (fMRI) scanning session. Here, our data consists of one graph per subject. Subjects can be partitioned by their *diagnostic status* (either ASD if diagnosed with autism spectrum disorder or TD if typically developed), and they can be grouped or selected by other attributes, such as *sex* (the only options being male and female), *age*, or *scanning modality* (eyes open or eyes closed).

The air transportation network data are taken from the website of the Bureau of Transportation Statistics (BTS). In the graphs representing these data, nodes correspond to airports in the United States, and weighted, directed edges correspond to volumes of passenger flows. Here, our data consists of one graph per *carrier class* and *month* from 2005 to 2020 (374 graphs in total).

The international trade network data are sourced from the World Integrated Trade Solution (WITS) provided by the World Bank. In the graphs representing these data, each node corresponds to a country (or similar unit), and each weighted, directed edge corresponds to the value of a trade flow. Here, our data consists of one graph per *product*

class (Animals, Vegetables, Food Products, Minerals, or Chemicals) and *month* from 1989 to 2018 (3 976 graphs in total).

We run Gragra on different subsets and splits of our datasets as shown in Table 3.2 and present a quantitative overview of our results in Figure 3.3. We observe that, in line with expectations derived from theory, *more graphs* or *graphs with more potential edges*, partitioned into *fewer groups*, generally yield *more patterns*.

Table 3.2: Real-world graph group data used in our experiments. n is the number of nodes, $[m]$ specifies the range of the number of edges per graph, k is the number of graph groups, and $[c_i]$ specifies the range of the group cardinalities. *TD* stands for *Typically Developed*, and *ASD* stands for *Autism Spectrum Disorder*. For the brain networks, which are sparsified during preprocessing, we use a minimum support of 2, and for the airline transportation networks and the international trade networks, we use an adaptive threshold of 0.1 times the cardinality of the smallest group in the experiment for sparsification. In all experiments, we use Vuong's test at a conservative significance level of 1×10^{-7} (or 1×10^{-5} when operating with less than 50 samples).

Dataset	**Description**	k	$[c_i]$
	Functional Brain Networks (undirected, unweighted) $n = 116$; $m \in [1\,320, 1\,348]$		
fbn-a	TD vs. ASD, age $[15, 20]$	2	$[116, 121]$
fbn-a1	ASD, age $[15, 20]$	1	$[116]$
fbn-c	TD vs. ASD, age ≤ 9	2	$[49, 52]$
fbn-c1	ASD, age ≤ 9	1	$[49]$
fbn-ac	TD vs. ASD × a vs. c	4	$[49, 121]$
fbn-e	TD vs. ASD, eyes closed	2	$[136, 158]$
fbn-e1	ASD, eyes closed	1	$[136]$
fbn-m	TD vs. ASD, males only	2	$[418, 420]$
fbn-m1	ASD, males only	1	$[420]$
	Air Transportation Networks (directed, weighted) $n = 300$; $m \in [335, 3\,533]$		
atn	all (2005–2020)	1	$[374]$
atn-m	major carriers	1	$[191]$
atn-n	national carriers	1	$[183]$
atn-c	carrier classes	2	$[183, 191]$
atn-q	quarters $[12, 3, 6, 9)$	4	$[92, 95]$
atn-y	four-year intervals	4	$[86, 96]$
	International Trade Networks (directed, weighted) $n = 250$; $m \in [256, 11\,415]$		
itn	all (1989–2018)	1	$[3\,976]$
itn-p	product class	5	$[210, 1\,530]$
itn-y	ten-year intervals	3	$[1\,314, 1\,332]$
itn-py	product class × intervals	15	$[70, 510]$
itn-a	animals	1	$[210]$
itn-ay	animals in intervals	3	$[70, 70]$
itn-v	vegetables	1	$[796]$
itn-vy	vegetables in intervals	3	$[247, 262]$
itn-f	food products	1	$[1\,137]$
itn-fy	food products in intervals	3	$[377, 380]$
itn-m	minerals	1	$[330]$
itn-my	mineral in intervals	3	$[110, 110]$
itn-c	chemicals	1	$[1\,530]$
itn-cy	chemicals in intervals	3	$[510, 510]$

Functional Brain Networks

Network neuroscience has emerged as a promising approach to understanding neurological disorders and diseases [16, 28, 58]. One of its fundamental questions is whether certain disorders are systematically associated with structural or functional connectivity alterations in the brain [77]. In particular, there is considerable uncertainty surrounding the neurological footprint of autism (and the delineation of its subtypes), and small sample sizes as well as covariates make many published findings hard to replicate [72, 90]. This calls for methods that can detect signal in the presence of considerable noise and heterogeneity, identifying connectivity patterns that are statistically significantly associated with one or more groups of brain networks.

Motivated by this application, we obtain graphs from preprocessed functional connectomes provided by the Autism Brain Imaging Data Exchange [38]. In these graphs, each node corresponds to one of the 116 *regions of interest* from the automated anatomical labeling atlas [AAL, 152], and each edge indicates relatively strong connectivity between two regions, as measured by their blood-oxygen-level dependent signal correlation during resting-state functional magnetic resonance imaging. To facilitate comparisons, the data is processed and grouped as described by Lanciano, Bonchi, and Gionis [98], but we remove the self-loops (corresponding to perfect self-correlations) that are present in their data.

We experiment with Gragra in four two-group settings (individuals with autism spectrum disorder [ASD] and typically developed controls [TD] in the categories *adolescents*, *children*, *eyes closed during scan*, and *males*), four one-group settings (autistic individuals in each category only), and one four-group setting (autistic and non-autistic children and adolescents), operating on graphs with $m \in [1\,320, 1\,348]$ edges and graph groups $\mathcal{G}_i$ with $c_i \in [49, 420]$ graphs. Our four-group experiment identifies significant overconnectivity across multiple brain regions as characteristic of ASD children versus all other groups, paralleling the neuroscience literature [134, 169]. However, as shown in Figure 3.4, most of the patterns we identify in the two-group setting yield similar information gains across both groups (left), and

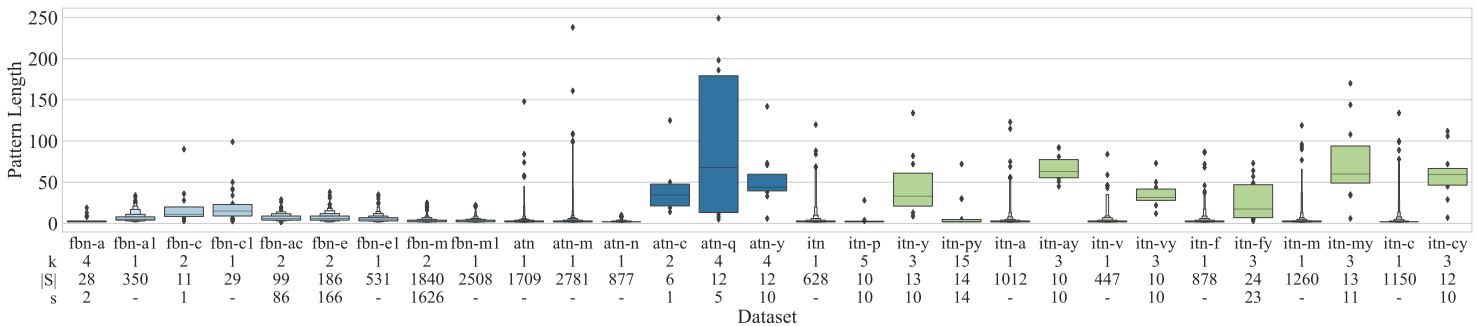


FIGURE 3.3: GRAGRA discovers long graph patterns in datasets with different numbers of graph groups. Here, we show the length distribution of the patterns identified in each of our experiments on real-world data, where each box corresponds to a dataset. The first number below a dataset identifier states the number of graph groups k in the dataset, the second number states the total number of patterns $|S|$, and the third number states the number of patterns s shared between at least two graph groups.

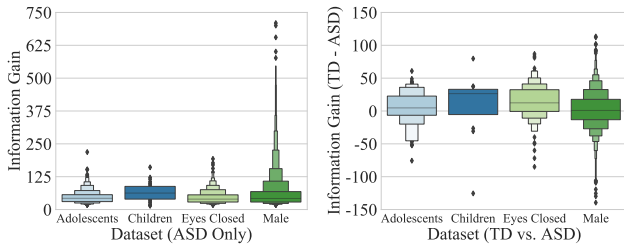


Figure 3.4: Gragra unveils shared and contrastive patterns in noisy and heterogeneous data. Here, we display the distribution of information gains per pattern in the one-group setting (left), and the distribution of information gain differences per pattern in the two-group setting (right), for our experiments on functional brain networks.

there is significant structure to be exploited even *within* individual groups (right). This indicates that the differences between autistic and non-autistic brains in the settings under study are rather subtle, and that there is considerable heterogeneity also in the one-group data. To explore this heterogeneity and delineate neurosubtypes of autism [cf. 78], our results could be used as inputs to multivariate subgroup discovery or clustering algorithms, where Gragra would effectively serve as a dimensionality reduction technique.

Air Transportation Networks

We obtain data on passenger flows between domestic airports in the United States for each month over the sixteen years from January 2005 to December 2020 from the website of the Bureau of Transportation Statistics [29]. Restricting our analysis to United States mainland airports and carriers classified as national (100 million to 1 billion USD revenue in the previous year) or major (over 1 billion USD revenue in the previous year), we create one air transportation network per year, month, and carrier class. To this end, for each year and month, we aggregate the passenger flows between two airports by carrier class and filter edges corresponding to fewer than 3 000 passengers, which leaves edges between $n = 300$ airports (identified by three-letter IATA codes). Excluding graphs with fewer than $n - 1 = 299$ edges, we arrive

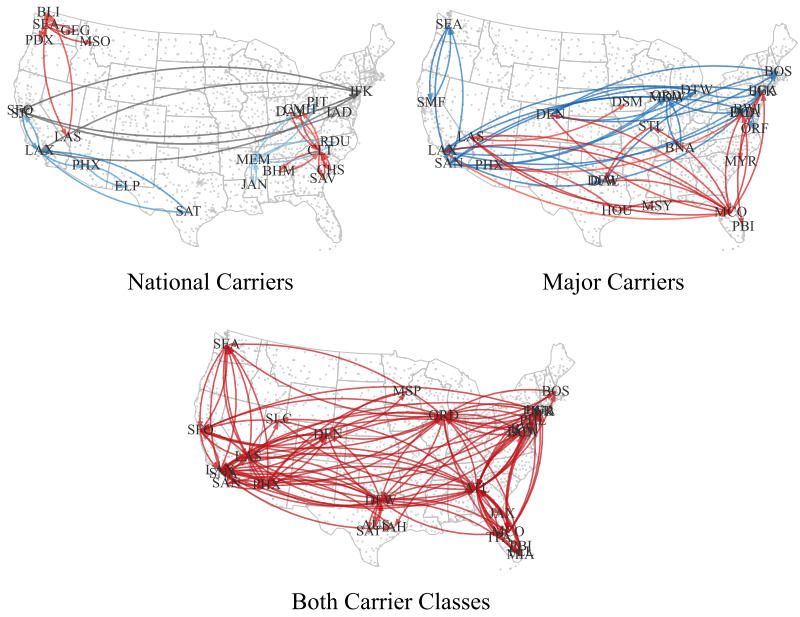


FIGURE 3.5: GRAGRA discovers large, meaningful graph patterns. Here, we depict some of the patterns discovered in the air transportation networks of national carriers (left, five patterns shown), major carriers (right, two patterns shown), and both carrier classes (bottom, one pattern shown). Gray nodes represent airports, and node labels identify airports contained in at least one of the displayed patterns by their three-letter IATA codes. Directed edges represent flight segments, and edge colors are proportional to their weight bins, following different color maps (reds, blues, or grays) where necessary to make them visually distinguishable. All drawn patterns are among the top fifteen in terms of information gain for their respective experiment, and the pattern in the middle is the top shared pattern, corresponding to the United States air transportation backbone.

at 374 graphs, whose edges we discretize into ten weight categories using equal-width binning.

We are interested in discovering patterns that are shared across all graphs, identifying structures of connected routes that are specific to individual carrier classes, and unveiling both seasonal and temporal trends. Therefore, we run GRAGRA in six different settings: on all graphs as one group, on the graphs corresponding to each carrier class separately, on all graphs with carrier classes as groups, on

all graphs with quarters as groups (starting from December to capture the winter holiday season), and on all graphs with consecutive four-year intervals as groups. Thus, our setup contains graphs with $m \in [335, 3\,533]$ edges and graph groups $\mathcal{G}_i$ with $c_i \in [86, 374]$ graphs. In Figure 3.5, we depict a subset of our results from the experiments involving the distinction between carrier classes. Gragra reveals an air transportation backbone jointly serviced by both carrier classes (middle), and it uncovers routes that are characteristically served by national or major carriers (left and right). Overall, we find that patterns corresponding to national carrier routes are often smaller and cover shorter distances than those corresponding to major carrier routes, mirroring the relatively smaller role of national carriers in the air traffic market.

International Trade Networks

We obtain data on international trade flows from the website of the World Integrated Trade Solution [192], for the thirty years from 1989 to 2018 (inclusive). The raw data correspond to exports of goods between (mostly) countries, classified using the Harmonized System at the four-digit level (HS-4), whose trade values we aggregate per (source, destination, HS-4 code) triple. For each year and HS-4 code, we construct one directed, weighted graph with (roughly) countries as nodes and exports as edges, discretizing the edge weights into ten categories using equal-width binning. We eliminate all trade entities above the country level but retain trade entities below the country level (and countries that do not exist anymore) if they have an ISO3 code. Restricting our attention to the WITS product groups *Animals*, *Vegetables*, *Food Products*, *Minerals*, and *Chemicals*, we arrive at 3 976 graphs with $n = 250$ nodes and at least $n - 1 = 249$ edges.

Leveraging the richness of our data, we ask not only what graph patterns are characteristic of international trade as a whole, but also what structures emerge when we group trade networks by product class, ten-year interval, or product class *and* ten-year interval. As Gragra allows us to inspect our data at different scales, we further investigate the trade patterns it unveils when considering each product

class separately, either treating all graphs from one product class as one group or splitting them by ten-year interval. Thus, we run our experiments on graphs with $m \in [256, 11\,415]$ edges and graph groups $\mathcal{G}_i$ with $c_i \in [70, 3\,976]$ graphs. In Figure 3.6, we illustrate five patterns discovered in the experiments that explore all graphs together, grouped by product class and ten-year interval. Although the input consists of fifteen classes, Gragra discovers not only meaningful patterns but meaningful patterns *with meaningful assignments* to graph groups which, as highlighted by the pattern labels in Figure 3.6, can be summarized succinctly. Across all experiments, we observe that the patterns yielding the largest information gains are often composed entirely of edges in the top two weight bins. This suggests that the ranking of exporter-importer pairs is most stable on the upper end of the trade-value spectrum, which aligns with interdisciplinary research findings that international trade is highly stratified [54, 110, 156].

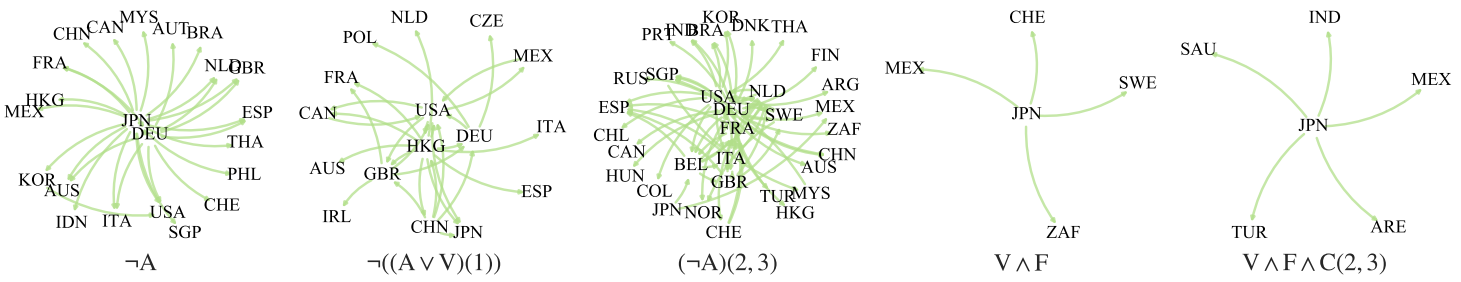


Figure 3.6: Gragra mines differential descriptions even when many graph groups are given as input. Here, we show the top five graph patterns identified in the international trade networks when split by product class and decade (fifteen graph groups in total). Nodes correspond to countries, which are represented by their ISO3 country codes. Directed edges correspond to trade flows between the countries, where the edge weights in all displayed patterns fall into the top weight bin. The patterns are labeled by rules identifying the graph groups in which they occur, with letters corresponding to the first letter of a product group, and numbers corresponding to the position of a ten-year interval. For example, the third pattern, labeled $(\neg A)(2, 3)$, occurs in all product classes except for *Animals*, in the second and the third ten-year interval, i.e., in $[99, 19)$.

3.7 Conclusion

We studied the *graph group analysis* problem: Given a set of graphs and a partition of this set into *graph groups*, succinctly summarize the commonalities and differences between graphs in the same group, between graphs in different groups, and between the relationships connecting the groups. We introduced Gragra as an algorithm to solve the problem, which uses maximum likelihood modeling, paired with a model selection criterion and a statistical test, to jointly discover a set of significant subgraphs, called graph patterns, and an assignment of these patterns to graph groups. In our experiments, we demonstrated that Gragra differentially describes synthetic and real-world graph groups, even when faced with heterogeneity, noise, or large group numbers. As a byproduct, we introduced two novel datasets of node-aligned graphs, which might be of independent interest to the graph mining community.

Our work is naturally limited. First, we model edge weights as categories, which works well for binned edge weights in practice but is theoretically dissatisfying. Hence, a natural enhancement of Gragra would be able to handle real edge weights, possibly using a maximum entropy model on its edge weight distribution. Second, Gragra is limited to groups of node-aligned graphs, and extending it to other graph types constitutes an open opportunity for future work. Third, we currently test all our graph patterns at the same significant level α. While this is theoretically defensible, given that we combine our statistical test with a model selection criterion, dynamically adjusting our alpha level could make our patterns even more powerful.

4
Discovering Significant Patterns under Sequential False Discovery Control

In Chapter 3, we sought to discover insightful patterns in *graph data*, for which we relied on a statistical test, and we concluded with the thought that dynamically adjusting the alpha level of that statistical test could further improve our results. Now, we embrace statistical significance in the setting from Chapter 2, i.e., we consider *binary tabular data*, but this time with the goal to discover *statistically* significant patterns.

We are interested in discovering those patterns from data with an empirical frequency that is significantly different than expected. To avoid spurious results, yet achieve high statistical power, we propose to *sequentially* control for false discoveries *during* the search. To avoid redundancy, we propose to update our expectations whenever we discover a significant pattern. To efficiently consider the exponentially-sized search space, we employ an easy-to-compute upper bound on significance, and propose an effective search strategy for sets of significant patterns. Through an extensive set of experiments on synthetic data, we show that our method, Spass, recovers the ground truth reliably, does so efficiently, and without redundancy. On real-world data, we show Spass works well on single and multiple groups as well as on low-dimensional and high-dimensional data, and through case studies, we demonstrate that it discovers meaningful results.

This chapter is based on the publication: Dalleiger and Vreeken [41].

4.1 Introduction

A cornerstone of many scientific problems is the discovery of statistically significant associations between features in data. In the biomedical domain, for example, researchers are interested in identifying combinations of genetic markers that are associated with specific phenotypes [106, 109, 198], studying combinations of mutations caused by cancer [179], or analyzing correlated markers that together indicate a high survival chance of a patient [150]. *Statistically significant pattern mining* is a branch of data mining in which we are after those patterns that are *statistically significant* with regard to some null hypothesis. Thus, it is particularly well-suited to meet the needs of many scientific domains.

A key issue plaguing significant pattern mining is the *multiple hypothesis testing problem*: If we test a single pattern for significance, the probability of falsely rejecting the null hypothesis is bounded by its p-value. This probability quickly converges to 1 as we test more hypotheses, however, and since the pattern search space is exponential in the number of binary features, we drown in spurious results unless we use some form of false discovery control. One option is to limit the risk of making at least one false discovery, also known as controlling the Family-Wise Error Rate (FWER), and another option is to limit the expected number of false discoveries, which is known as controlling the False Discovery Rate (FDR). Most work in the field focuses on finding a good balance between *statistical power* and *computational efficiency*.

Although recent work achieves impressive results, it falls short when it comes to reporting succinctly and without redundancy. To illustrate this problem, we run three recent statistically significant pattern miners, Lamp [175], WYlight [108], and SPuManTE [141], on synthetic two-group data using only 100 ground truth patterns, randomly associated to each group. The higher the group association probability, the more patterns participate in generating the group. At 100%, for example, all 50 patterns are used to generate each group (see Section 4.6 for more details). In Figure 4.1, we show the number of

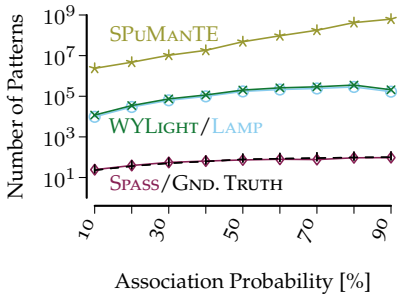


Figure 4.1: Spass recovers the ground truth where competitors struggle. We show the number of *statistically significant* patterns discovered at an FWER of 0.05 on two-group data over 500 unique items, where we vary the probability of associating 100 ground truth patterns with its groups.

patterns discovered by Lamp, WYLight, SPuManTE, and our method at a significance level of 0.05. There, Lamp, WYLight, and SPuManTE identify orders of magnitude more patterns as significant than we originally used to generate the data—although technically not incorrectly, since subsets or combinations of ground-truth patterns might also be significant. However, these redundant results drown the analyst in patterns.

To achieve concise and informative, rather than redundant results, we propose to test patterns for significance against our expectation, based on the patterns we have discovered so far. To prevent spurious results, yet achieve high statistical power, we sequentially control for either family-wise error rate or false discovery rate. That is, we iteratively adjust the significance level α during the search, factoring in what part of the space we have already explored and what hypotheses we have rejected. Our method, Spass (Significant Pattern ASSociations), automatically associates patterns to those groups for which they are significant, thereby immediately exposing the similarities and differences between the groups. This allows us to handle data with one or more groups, while existing methods can only handle data with one or two groups.

Through an extensive set of experiments, we show that Spass performs well in practice. While its competitors drown the analyst in large numbers of highly redundant patterns, we demonstrate that Spass reli-

ably recovers the ground truth in synthetic data and discovers succinct and non-redundant patterns in real-world data. In two detailed case studies on real-world data, we illustrate that the patterns identified by Spass are also meaningful. Furthermore, Spass is very fast, taking only seconds up to a few minutes in our experiments where competitors take hours, days, or even weeks.

Our main contributions are that we

1. suggest to iteratively test for significance against a probabilistic model of the data based on our most recent knowledge of the data,
2. propose a novel sequential FWER control and introduce the first pattern mining procedure under sequential online FDR control,
3. show how to sequentially control for either FWER or FDR,
4. introduce the Spass algorithm to efficiently discover non-redundant sets of statistically significant patterns using an easy-to-compute Chernoff bound, and
5. provide an extensive empirical evaluation on synthetic and real-world data.

The remainder of this chapter is structured as follows. After settling the preliminaries in Section 4.2, we state our problem formally in Section 4.3. Next, we introduce and analyze our method Spass in Section 4.4. Related work is discussed in Section 4.5, and we empirically evaluate Spass in Section 4.6. We discuss the merits of our method in Section 4.7, and conclude the chapter in Section 4.8.

4.2 Preliminaries

As in Chapter 2, we consider binary tabular data. Therefore, our notation largely follows Section 2.2. In brief, we write 2^A for the powerset of any finite set A, and $\binom{A}{k}$ for the set of all subsets of A of size $k \in \mathbb{N}$. The set $A \bigtriangleup B$ is the symmetric difference of A and B. For any $n \in \mathbb{N}$, we write $[n] = \{1, 2, \ldots, n\}$. The indicator function is $\mathbf{1}$. All logarithms are to base 2, and by convention, we use $0 \log 0 = 0$.

We consider binary data X over d features $\mathcal{I}$. A dataset X is a multiset of n samples from the set $\Omega = 2^{\mathcal{I}}$, of all possible samples. Like in Section 2.2, for a given partitioning of $\Pi \in \omega(X)$, we denote

the $k \geq 1$ groups in Π by $\{X_1, \ldots, X_k\}$ and let $n_i = |X_i|$. Our method requires an underlying probabilistic model, for which we again choose the maximum entropy distribution introduced in Chapter 2.

4.3 Significant Pattern Sets

In this section, we state our problem. We first do so informally, after which we move to the statistical test for one hypothesis, describe its efficient inference, and introduce our sequential hypothesis-testing procedure.

4.3.1 The Problem, Informally

Our goal is to discover those patterns whose empirical frequencies in the data differ significantly from what we expect, based on what we already know about the data. We strive to do this for datasets with one or multiple groups over the same set of binary features, such that we find not only patterns that are distributed significantly differently in general but also patterns that are distributed significantly differently in one or multiple groups.

We explicitly seek to prevent redundant results, and hence require that every reported pattern is significant in light of all previously discovered patterns. This formulation has the benefit that it allows us to *sequentially* control for false discoveries by adjusting the significance threshold during the search, based on what part of the search space we have considered so far.

Existing statistical pattern-mining approaches report every significant pattern, often including subsets or combinations of true patterns, which introduces redundancy. The key idea of our approach is that we discover non-redundant results by testing each pattern for significance against a model of the data based on prior discoveries, and do so using an appropriately adjusted significance level.

A bit more formally, our goal is to discover one pattern set S_i for each group X_i, such that the empirical frequency $q_{S_i}(x)$ of each pattern $x \in S_i$ diverges significantly from our expected frequency $p_{S_i \setminus \{x\}}(x)$

based on patterns already accepted prior to x, while controlling for false discoveries.

With this intuition in mind, we next describe our probabilistic model and the statistical test for *one* hypothesis. Afterwards, we show how to sequentially control for false discoveries when testing multiple hypotheses using either family-wise error rate (FWER) or false discovery rate (FDR).

4.3.2 Testing for Significance

To infer an expected frequency $p_{S_i}(x)$, we need a probability distribution p. We again choose the maximum entropy distribution introduced in Section 2.3, which models the data precisely and does not introduce any bias. Accordingly, we rely on inferring the expected frequency of a maximum entropy distribution (cf. Eq. (2.3.3)), and define our expectation as $p_S(x) = \mathbb{E}_f[x \mid S]$.

Formally, we state the *null hypothesis* that the distribution of an $x \subseteq \mathcal{I}$ in group X_i follows the expectation p given $S_i \subseteq \Omega$ as

$$H: \quad q_{X_i}(x) = p_{S_i}(x) \ ,$$

and the *alternative hypothesis* that it is distributed differently as

$$H^a: \quad q_{X_i}(x) \neq p_{S_i}(x) \ .$$

A pattern x either occurs in a sample in X_i or it does not, and under the null hypothesis, it is hence Bernoulli distributed with success probability $p_{S_i}(x)$. By convention, we assume that each sample in X_i is independently drawn, such that the number of samples in which x occurs is binomially distributed. Under the null hypothesis, the p-value $\mathbb{P}[H]$ of observing a pattern x with a more extreme frequency $q_{X_i}(x)$ than our expectation $p_{S_i}(x)$ is

$$\mathbb{P}\left[n_x \geq \hat{n}_x \mid H\right] \ ,$$

where $n_x = |X_i| \cdot q_{X_i}(x)$ is the *observed* number of data points that supports x, and $\hat{n}_x = |X_i| \cdot p_{S_i}(x)$ is the *expected* number of such data points.

To infer these p-values exactly, we can use the binomial cumulative distribution function

$$F(s, n; p) = \sum_{k=s}^{n} \binom{n}{k} p^k (1-p)^{n-k} \ , \tag{4.3.1}$$

for the number of successes s, number of trials n, and success probability p. More precisely, if $q_{X_i}(x) \geq p_{S_i}(x)$, we can infer $\mathbb{P}[H]$ by computing $F(n_x, |X_i|\,; p_{S_i}(x))$, or else, we do so using $F(0, n_x; p_{S_i}(x))$. Although mathematically convenient, as we may have to infer the CDF exponentially often, computing F exactly during our search is impractical. We therefore propose to approximate F using the easy-to-compute Chernoff bound [33],

$$F(n \cdot a, n; b) \leq \exp\left[-nD(a\|b)\right] \ ,$$

where $D(a\|b)$ is the Kullback-Leibler divergence

$$a \log a/b + (1-b) \log(1-a)/(1-b)$$

of the two Bernoulli distributions a and b. To illustrate how well the Chernoff bound approximates the binomial CDF in comparison to the popular standard normal approximation, we show p-values for 10 and $1\,000$ samples for a success probability of 0.5 in Figure 4.2. There, we see that the Chernoff bound tightly approximates exactly computed p-values, even for few samples.

4.3.3 Controlling for False Discoveries

If we test a single hypothesis, the probability of falsely rejecting the null hypothesis H is bounded by its p-value $\mathbb{P}[H]$. As we test more hypotheses, the probability of falsely rejecting at least one null hypothesis converges to 1—that is, unless we carefully control for testing multiple hypotheses. We consider two approaches to false discovery control, namely, one targeting the family-wise error rate and one targeting the false discovery rate. Both have in common that, rather than testing each hypothesis at the same significance level α, they test hypotheses

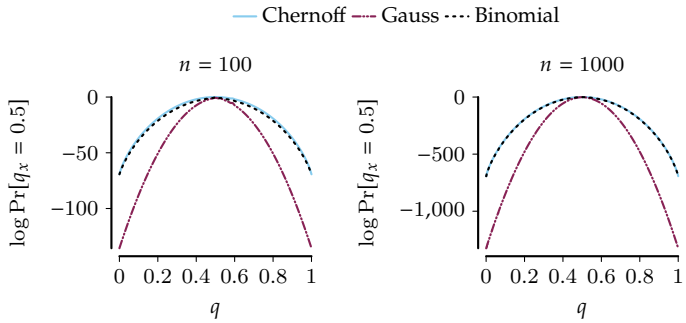


Figure 4.2: The Chernoff bound closely approximates the binomial CDF. For the fixed probability $p(x)$ of 0.5, we show p-values for the Chernoff bound, the Gaussian approximation, and the exact binomial CDF over 100 (left) and $1\,000$ (right) samples.

at *adjusted* significance levels $\alpha_t < \alpha$. In a nutshell, in both cases, we consider a sequence of hypotheses

$$H_1, H_2, \dots ,$$

for which we compute the corresponding sequence of p-values

$$\mathbb{P}[H_1], \mathbb{P}[H_2], \dots .$$

We decide to reject the t^{th} hypothesis H_t if its p-value $\mathbb{P}[H_t]$ is less than the *adjusted* test level α_t, i.e.,

$$\mathbb{P}[H_t] < \alpha_t ,$$

and denote the set of all rejections as $\mathcal{R} = \{H_t \in \mathcal{H} \mid \mathbb{P}[H_t] < \alpha_t\}$, where $\mathcal{H}$ is the set of all hypotheses. Regardless of how we control α_t, we ideally want to maximize the number of true discoveries (statistical power), also known as the *true discovery rate* (TDR)

$$\mathbb{E}\left[\frac{|\mathcal{R} \cap \mathcal{H}^a|}{\max\{1, |\mathcal{H}^a|\}}\right] ,$$

where $\mathcal{H}^a = \{H \in \mathcal{H} \mid H^a = 1\}$ is the unknown set of truly alternative hypotheses. In the following, we discuss how to determine a test

level sequence α_t that achieves a high TDR while controlling for false discoveries, starting with the conservative family-wise error rate and then moving on to the less conservative false discovery rate.

Controlling FWER We start with the adjustment of the significance levels α_t to guarantee a FWER of at most α. The *Family-Wise Error Rate*, or FWER for short, is the probability

$$\mathbb{P}[|\mathcal{R} \cap \mathcal{H}_0|] > 0$$

of making at least one false discovery, where $\mathcal{H}_0 = \{H \in \mathcal{H} \mid H = 1\}$ is the unknown set of all hypotheses that are truly null. We can keep the FWER below α by testing against an adjusted significance threshold $\alpha_t = \alpha/N$, where we simply divide α by the number N of hypotheses in $\mathcal{H}$. This is known as *Bonferroni correction* [25]. While Bonferroni correction works well when testing relatively few hypotheses, in our case, $N = k \cdot 2^m$ is exponential in m, and hence, for any non-trivial value of m, the adjusted values α_t will be so low that the probability of a true discovery is (almost) zero.

In statistically significant pattern mining, one common approach to increase the TDR is by outright excluding hypotheses if their *minimally attainable p-value* is above the significance threshold [171]. This is *Tarone's exclusion principle* [128]. Since the minimally attainable p-value in our case shrinks exponentially with number of samples in a group (cf. the infimum of Eq. (4.3.1)), it becomes so small that we cannot exclude many hypotheses in advance. We can, however, exploit the fact that we typically do not test all patterns but rather a much smaller set C of candidate patterns. Hence, it suffices to adjust α by the size of C, rather than N, since $|C| \ll N$. Unfortunately, we do not know C in advance. But fortunately, we do know how we *generate* C. We, therefore, make our significance level adjustment *search-space aware* [17, 186, 188]. That is, we *sequentially* adjust the significance levels α_t while we iteratively generate C.

To do so, we need to impose structure on the search space. We propose to organize the hypotheses (i.e. patterns) as a lattice, such that layer l contains all patterns of length l. If we now search for significant

patterns layer by layer, we only have to correct for up to and including the current layer l, which is at most the sum of all binomials up to l. While easy to compute for small l, for larger values, this sum is computationally costly, and hence, we resort to the upper bound

$$\sum_{k=1}^{l} \binom{m}{k} < \sum_{k=1}^{l} \frac{m^k}{k!} = \sum_{k=1}^{l} \frac{(m/l)^k l^k}{k!} \leq (m/l)^l \sum_{k=1}^{\infty} \frac{l^k}{k!} = \mathrm{e}^l (m/l)^l \ . \quad (4.3.2)$$

To obtain the adjustment factor we need for the tth hypothesis, we multiply the right-hand side of Eq. (4.3.2) with the number of groups k. We summarize the above in the following lemma.

Lemma 4.1. For any sequence of p-values, we control for the FWER by adjusting the test levels, for the t^{th} hypothesis using

$$\alpha_t = \min_{s<t} \left\{ \alpha_s, \ \alpha[k \cdot \mathrm{e}^l (m/l)^l]^{-1} \right\} \ , \quad (4.3.3)$$

where l is the highest layer in the search lattice explored so far, $m = |\mathcal{I}|$ is the number of features which coincides with the highest lattice level, and k is the number of groups under consideration.

Proof. At each level $l \in \{1, \ldots, m\}$, we adjust for all possible hypotheses up to layer l, which grows to at most m. By summing Eq. (4.3.2) up to m, we achieve equality with Bonferroni correction. □

Although many domains require FWER, there are problems that do not need such strict error control. In these cases, we therefore control for the less conservative FDR, described next.

Controlling FDR The *False Discovery Rate* [21], or FDR for short, is an alternative approach to false discovery control. To permit a higher statistical power than FWER, the FDR is controls for the *expected number* of false discoveries

$$\mathbb{E}\left[\frac{|\mathcal{R} \cap \mathcal{H}_0|}{\max\{1, |\mathcal{R}|\}} \right] ,$$

rather than the probability of at least one false discovery.

To ensure an FDR of at most α, we can determine α_t using so-called generalized α-investing (GAI) rules [6]. These rules "invest" a fraction of our available "α-budget" in each significance test we perform, thus decreasing the available α-budget, and reward each discovery by increasing the available α-budget. In short, we start with a budget of $0 < w_0 < \alpha$, decrease the budget when testing the tth hypothesis with a penalty ϕ_t, and increase the budget with a reward ψ_t when we reject it. Thus, we can continue testing as long as we make discoveries. Formally, our budget develops as

$$w_{t+1} \leftarrow w_t - \phi_t + \mathbf{1}[\mathbb{P}[H_t] < \alpha_t] \cdot \psi_t \,. \tag{4.3.4}$$

Since our p-values are statistically dependent, and we seek high statistical power, we employ a variant of the LORD-update rules proposed by Javanmard and Montanari [84]. We start with an initial budget of $w_0 = \alpha/2$. For every discovery, we receive a constant reward $\psi_t = \alpha - w_0$. To prevent that we use all available budget on a single hypothesis, we set the penalty ϕ_t and the level α_t to a fraction

$$\alpha_t \leftarrow \gamma_t \cdot w_\tau$$

of the budget w_τ available at the most recent discovery time

$$\tau = \arg\max_{s<t} \mathbf{1}\{\mathbb{P}[H_s] < \alpha_s\} = 1 \,,$$

using a non-increasing sequence $(\gamma_t)_{t\geq 1}$ as the fraction of our budget w_t that we invest into the test in iteration t. To choose such a sequence $(\gamma_t)_{t\geq 1}$ for *arbitrarily dependent* p-values, we resort to Thm. 3.7 from Javanmard and Montanari [84]. In essence, this theorem states that any non-increasing sequence $(\gamma_t)_{t\geq 1}$ guarantees a bounded FDR if

$$\sum_{t}^{\infty} \gamma_t(1 + \log t) \leq \alpha/w_0 \,,$$

holds. In particular, this is true for the sequence

$$\gamma_t = \frac{6}{t^2\pi^2}\frac{\alpha/w_0}{(1+\log t)}$$

which we summarize in the following lemma.

Lemma 4.2. For $\gamma_t = \frac{6}{t^2\pi^2}\frac{\alpha/w_0}{(1+\log t)}$, the generalized α-investing rules described above control FDR for arbitrarily dependent p-values.

Proof. By substituting γ_t in Thm. 3.7 from [84], we observe that the factor $1 + \log t$ cancels out. Since $\sum_t^\infty \frac{6}{t^2\pi^2}$ converges to 1, the series converges exactly to α/w_0. □

4.4 Algorithm

Now, we introduce our algorithm Spass for efficiently discovering significant, non-redundant pattern sets under false discovery control. We give the pseudocode of Spass as Algorithm 4.1.

Starting with an empty pattern set S_i for each group X_i (l. 1) and an initial search space C that contains all itemsets of length two (l. 2), we set the counter tallying significance tests to 1 (l. 3). Then, we expand the search space C using the Search algorithm, detailed below, selecting that candidate

$$\hat{x} \leftarrow \arg\min_{x \in C} \mathbb{E}_{S_i}[p_{S_i}(x) = q_{X_i}(x)] \tag{4.4.1}$$

which has the lowest expected chance (l. 5)

$$\mathbb{E}_{S_i}[p_{S_i}(x) = q_{X_i}(x)] = \sum_{i=1}^{m} \mathbb{P}[p_{S_i}(x) = q_{X_i}(x)]$$

of resulting in false discoveries, across all distributions. We test the significance of $\hat{x}$ (l. 9) for every group X_i (l. 6) against a significance level that is appropriately adjusted according to either FWER or FDR (ll. 7–8). If the candidate is significant (l. 10), we reject the null hypothesis, add $\hat{x}$ to S_i, and re-infer the distribution $p_{S_i}(\cdot)$ (ll. 11–12). We iterate until convergence, and finally return the k pattern sets (l. 13).

Algorithm 4.1: **Spass**

Input: groups $X_1, \ldots, X_k$, test level $\alpha \in [0, 1]$
Output: patterns sets $S_1, \ldots, S_k$

$S_i \leftarrow \emptyset \; \forall i \in \{1, \ldots, k\}$
$C \leftarrow \{x \subseteq \mathcal{I} \mid |x| = 2\}$
$t \leftarrow 1$
while $C \neq \emptyset$
    $\hat{x}, C \leftarrow$ Search(C)
    foreach group X_i **do**
        $t \leftarrow t + 1$
        adjust test level $\alpha_t \leftarrow$ [Eq. (4.3.3) or Eq. (4.3.4)]
        hypothesize $H_t : p_{S_i}(\hat{x}) = q_{X_i}(\hat{x})$
        if $\mathbb{P}[H_t] < \alpha_t$
            $S_i \leftarrow S_i \cup \{\hat{x}\}$
            estimate coefficients for p_{S_i}
return $S_1, \ldots, S_k$

To identify the next pattern to test, we use Algorithm 4.2. Given the current search space C, we first find the most promising candidate $\hat{x} \in C$ using Eq. (4.4.1) (l. 1). We then expand C with all combinations of $\hat{x}$ and singletons $y \in \mathcal{I}$ (l. 2). Note that this corresponds to exploring (part of) layer $l + 1$ of the lattice, where $l = |\hat{x}|$ is the layer which $\hat{x}$ resides. If there exists an x in the now-expanded search space C that is better than $\hat{x}$, we recurse (ll. 3–4), and otherwise, we return the best candidate $\hat{x}$ and the search space C (ll. 5–6) we explored so far.

Algorithm 4.2: **Search**

Input: search space $C \subseteq \Omega$
Output: candidate $\hat{x}$ and expanded search space C

1 $\hat{x} \leftarrow \arg\min_{x \in C} \mathbb{E}_{S_i}[p_{S_i}(x) = q_{X_i}(x)]$
2 $C \leftarrow C \cup \{\hat{x} \cup \{y\} \mid y \in \mathcal{I}\}$
3 **if** $\min_{x \in C} \mathbb{E}_{S_i}[p_{S_i}(x) = q_{X_i}(x)] < \mathbb{E}_{S_i}[p_{S_i}(\hat{x}) = q_{X_i}(\hat{x})]$
4   **return** Search(C)
5 **else**
6   **return** $\hat{x}, C \setminus \hat{x}$

The computational complexity of Spass depends on the size of C, which grows binomially with each layer of expansion, and can reach up to 2^m elements. Assuming that the complexity of inferring the expectations p is bounded by a constant, the worst-case complexity is $O(2^m)$. Algorithm 4.1 has a complexity of $O(e^l(m/l)^l)$ after reaching the lth lattice layer, and in realistic applications, we never explore the entire lattice. As we do not expand layers fully either, Spass is still more efficient in practice.

4.5 Related Work

Pattern mining is arguably one of the most important and well-studied areas of data mining. Traditional approaches, such as *frequent itemset mining* [5], aim for completeness, and return all patterns that satisfy some user-defined constraints. By scoring patterns individually, the results of traditional methods are typically very large, highly redundant, and mostly spurious [3].

Pattern set mining aims to search only a small and non-redundant set of patterns that together generalize the data well. Typical quality measures include probabilistic objective functions [60] or information-theoretic scores [181], and algorithms have been used for characterizing data with one [57] or multiple groups [27, 43] (cf. Chapter 2). Although these methods discover succinct, non-redundant sets of patterns that have been proven useful, the results come without statistical guarantees, which bars their application in critical domains, such as genetics [106, 109, 179, 198], survival analysis [150], or network analysis [166].

Significant pattern mining provides statistical guarantees by using statistical tests to prune out spurious results. There exist many significance tests, and hence almost as many dedicated statistically significant pattern mining methods, e.g., for Fisher's exact [70, 175], Mann-Whitney-Wilcoxon [175], conditional permutation [198], Westfall-Young permutation [108, 142], Cochran–Mantel–Haenszel [138], Barnard's unconditional [141], or the Likelihood ratio test [167]. Each of these methods corrects for multiple hypothesis testing mostly targeting FWER

and using Bonferroni [25] correction. Some methods use Tarone's exclusion principle [171] to increase the statistical power. Another approach to cope with the low statistical power exhibited under Bonferroni correction is to make the adjustment "search-aware" [17, 186, 188] and directly adjust it, without necessarily knowing the total number of hypotheses to adjust for in advance. A search-aware significance level adjustment is also used for the search of non-redundant top-k statistically tested-to-be informative patterns [190]. Although these methods rigorously control for false discoveries, they still test against a static null hypothesis. As a result, they report every significant pattern, and consequently, they tend to discover many and highly redundant results—often orders of magnitude more than there are samples in the data.

Our goal with Spass is to discover concise, non-redundant sets of statistically significant patterns. Here, we combine the best of pattern set mining and statistically significant pattern mining. Our approach is unique in that it marries sophisticated probabilistic modelling to rigorous statistical testing, while accounting for the multiple hypothesis testing problem using a sequential and search-aware significance level adjustment that can target either FWER or FDR.

4.6 Experiments

We implement Spass in C++, and run experiments on an Intel Xeon E5-2643 CPU, reporting wall clock time. To differentiate between FWER and FDR control, we write Spass-fwer and Spass-fdr, respectively. We compare Spass to three methods for significant pattern mining, Lamp [175], WYlight [108] and SPuManTE [141]; two methods for non-redundant pattern set mining, mtv [119] and Desc [43]; and one statistically non-redundant pattern miner, opus [190]. All our competitors represent the state of the art in their respective fields. We report results at a significance level α of 0.05.

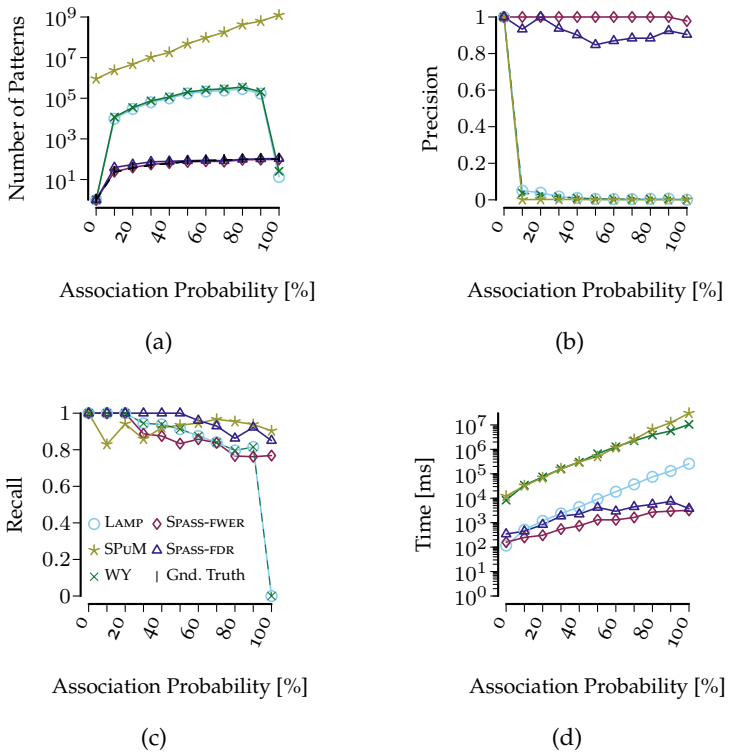


FIGURE 4.3: Our method efficiently recovers the ground truth with high precision and recall. Given are (a) number of significant discoveries, (b) precision, (c) recall, and (d) runtime, for LAMP, WYLIGHT, SPUMANTE, and our method, SPASS, on synthetic data over 500 unique items, with two groups of 5 000 samples each, in which we plant up to 100 ground-truth patterns overall. We vary the association probability p_a by which we independently associate patterns to groups; for $p_a = 0$, no patterns are planted, while for $p_a = 1$, every pattern is present in both groups.

4.6.1 SYNTHETIC DATA

To validate that SPASS recovers true patterns, we start by evaluating the algorithm on two-group data with known ground truth. To this end, we generate synthetic data as follows. First, we sample 100 random patterns of up to 5 items from an alphabet of $|\mathcal{I}| = 500$ items and insert them into a ground-truth pattern set S_i^* with an "association" probability varying in between 0% (no patterns planted at all) and 100% (all patterns are shared among all S_i^*). Then, we randomly draw 5 000 samples for each group X_i, in such a way that the ground truth

patterns $x \in S_i^*$ all have a randomly chosen frequency between 15% and 30%. Afterwards, we add additive noise of 5% and destructive noise of 1%. To account for random fluctuations, we average over 10 samples per 10% increment in probability.

We run the (two-group capable) significant pattern miners Lamp, WYlight, SPuManTE, and Spass on each dataset and report the average number of statistically significant discoveries in Figure 4.3a. At 0% association probability (i.e., no patterns, pure noise) SPuManTE is the only method that wrongfully discovers patterns. Across the board, we see that Lamp, WYlight, and SPuManTE all report orders of magnitude more patterns as significant than the number of patterns used to generate the data. As subsets or combinations of significant patterns are often significant as well, this is not incorrect per se. Spass, in contrast, almost matches the ground truth in number. At 100% association probability, there are no contrasting patterns and only shared patterns. Lamp and WYlight only identify that there are almost no contrasting patterns, whereas Spass correctly identifies that all patterns are shared among all groups.

To assess the quality of the discovered patterns, we measure precision and recall with respect to the ground truth as follows. We match each discovered pattern x with the best-matching ground truth pattern y in terms of set similarity $|x \cap y| / |x \cup y|$. Reporting precision in Figure 4.3b and recall in Figure 4.3c, we see that all methods obtain good recall, but due to their huge result sets, Lamp, WYlight, and SPuManTE have very low precision. The sequential redundancy control of Spass, however, prevents the exponential growth in the cardinality of the output, and consequently, Spass is both precise and often orders of magnitude faster than the competition (see Figure 4.3d).

High-Dimensional Synthetic Data Having ensured that Spass results in non-redundant discoveries under either FDR or FWER control, we investigate how much of a difference FWER and FDR can make on high-dimensional synthetic data

From Eq. (4.3.3), it follows that for a very large number of features m or a particularly high search depth l, FWER control becomes very

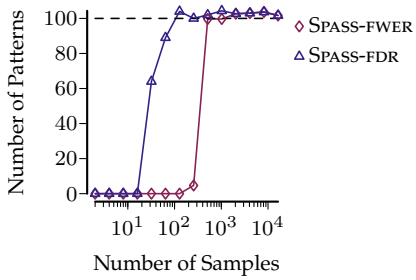


Figure 4.4: FWER is more conservative than FDR. We show the number of significant patterns discovered by Spass controlling for FWER (diamond) or FDR (triangle), respectively, at $\alpha = 0.05$, on synthetic data over $20\,000$ items with 100 ground-truth patterns (dashed line) while varying the number of samples.

strict. This means that a high dimensionality or very large patterns are particularly challenging. We generate data as above, but now over an alphabet $\mathcal{I}$ of $20\,000$ items in which we plant 100 random patterns. We run Spass with FWER resp. FDR on data with varying numbers of samples, and report the number of significant discoveries in Figure 4.4. We see that both variants converge to the correct number of patterns, but Spass-fdr does so much more quickly, requiring between one and two orders of magnitude less data. As expected, FDR is better-suited than FWER for high-dimensional data.

4.6.2 Real-World Data

Now that we have validated that Spass works well on synthetic data, we explore how it performs in a wide range of real-world scenarios.

All datasets used in our experiments are publicly available. We obtained genomics data from *The Cancer Genome Atlas Program*,[1] and binarized it using a specialized method for gene-expression data [61]. Furthermore, we took *Mushroom*, and *Pumsb** from the Itemset Mining Dataset Repository.[2] The *AG News* dataset consists of news articles from 4 categories,[3] and the *CORD 19* dataset consists of abstracts from

[1] cancer.gov/tcga
[2] fimi.ua.ac.be/data
[3] di.unipi.it/~gulli/AG_corpus_of_news_articles

the CORD 19 open research dataset.[4] We lemmatized the *AG News*, *CORD 19*, *IMDb*[5], and *ArXiv* datasets and removed stop words and words with a frequency below 0.1%. All the remaining datasets are from the UCI Machine Learning Repository[6] or from the LIBSVM repository.[7] To reduce the number of features of the *Instacart* dataset, we combined products from the same category, e.g., we merged Spumante with Cremant to achieve the Champagne meta category.[8] We binarized each real-valued feature by binning it into 5 bins of equal width, and we mapped each categorical and ordinal attribute to multiple binary features, which is often referred to as "one-to-k" or "one-hot" encoding. In Table 4.1, we provide basic statistics for the processed data.

Without access to the ground truth, we cannot compute precision and recall. We can, however, assess the number of discoveries and runtime of Spass relative to its competitors Lamp, WYlight, and SPuManTE, which we report in Figure 4.5.

In the left panel in Figure 4.5, we see that the competitors deem orders of magnitude more patterns as significant than Spass. Furthermore, we find that our method discovers fewer patterns when controlling for the more conservative FWER instead of the FDR. From the right panel in Figure 4.5, we observe that this tendency is reflected in the runtime of Spass-fwer, which is typically lower than that of Spass-fdr. Regardless of the control method, Spass is also almost always faster than its competitors.

Having ascertained that Spass efficiently discovers concise pattern sets from real-world data, we turn to case studies to answer three specific questions:

1. Does Spass work on *high-dimensional* real-world data?
2. Does Spass discover *meaningful* patterns in real-world data?
3. Can Spass compete with the state of the art in statistical pattern mining on *one-group* real-world data?

[4] allenai.org/data/cord-19
[5] ai.stanford.edu/~amaas/data/sentiment
[6] archive.ics.uci.edu/ml
[7] csie.ntu.edu.tw/~cjlin/libsvmtools/datasets
[8] instacart.com/datasets/grocery-shopping-2017

Table 4.1: We show the number of data points, the number of features, the average number of 1s per row, the overall density, and the number of groups k for the datasets used in our experiments.

Dataset	$\lvert X\rvert$	$\dim X$	Avg. Row	Density	k
Higgs	11 000 000	247	28.00 ± 0.00	0.1134	2
SUSY	5 000 000	178	18.00 ± 0.00	0.1011	2
Instacart	2 620 570	1 235	3.14 ± 2.18	0.0025	1
KDD Cup 99	1 000 000	135	16.00 ± 0.00	0.1185	1
Covtype	581 012	64	11.95 ± 0.23	0.1866	2
RNA	271 617	16	8.00 ± 0.00	0.5000	2
News	127 600	11 489	13.63 ± 4.05	0.0012	4
IJCNN	91 701	34	13.00 ± 0.00	0.3824	2
IMDb	49 969	8 125	63.95 ± 42.56	0.0079	2
Pumsb*	49 046	2 088	50.48 ± 1.98	0.0242	1
CORD-19	32 907	2 648	47.63 ± 23.87	0.0180	1
Adults	32 561	123	13.87 ± 0.48	0.1128	2
Mushroom	8 124	117	22.00 ± 0.00	0.1880	2
Breast Cancer	7 325	397	11.67 ± 13.06	0.0294	2
Metabric	1 981	124	32.32 ± 1.03	0.2606	2
Breast	1 218	20 530	3 036.89 ± 359.03	0.1863	1
Lung	1 129	20 530	3 378.75 ± 318.66	0.2043	2
Kidney	1 020	20 530	3 325.43 ± 242.96	0.2097	3
Kidney Clear	606	20 530	3 496.35 ± 371.08	0.2291	1
Lung Adeno.	576	20 530	3 053.31 ± 347.88	0.1932	1
Lung Squamous	553	20 530	3 146.87 ± 333.37	0.1972	1
Brain	530	20 530	3 099.68 ± 371.75	0.2146	1
Endo & Ovo	509	20 530	3 681.89 ± 290.47	0.2303	2
Ovarian	308	20 530	3 063.36 ± 307.32	0.2025	1
Uterine	57	20 530	3 224.40 ± 274.13	0.2253	1

High-Dimensional Real-World Data

To verify whether Spass works on high-dimensional real-world datasets, we consider ten genomics datasets concerning *Ovarian*, *Lung*, *Kidney*, *Brain*, and *Breast* cancer.[9] Together, these data span a wide range of different sizes, numbers of groups, and numbers of samples, with the shared trait that they are high-dimensional. We run Lamp, WYlight, SPuManTE, and Spass on each dataset, but Lamp, WYlight, and SPuManTE do not report any discoveries. In Figure 4.6, we report the number of discoveries by Spass after 2 hours of runtime. Here, as in our experiments on high-dimensional synthetic data,

[9] cancer.gov/tcga

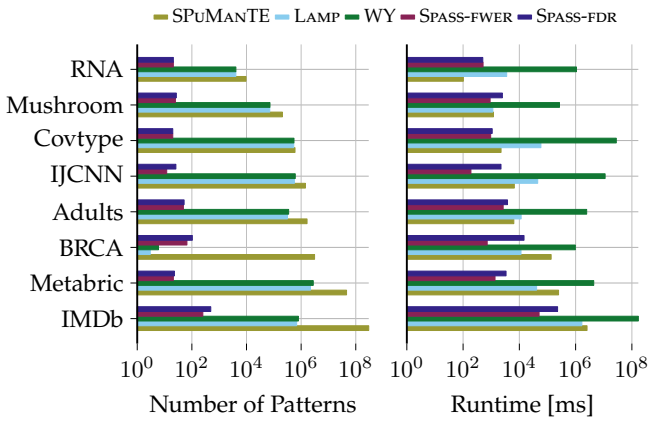


Figure 4.5: Unlike its competitors, Spass efficiently discovers concise pattern sets. We show the number of significant discoveries (left) and runtime needed (right) by SPuManTE, WYlight, Lamp and Spass for eight real-world, two-group datasets.

we see that FWER is much more stringent than FDR. For the *Lung A.* dataset, Spass-fwer only discovers 3 significant patterns—its highest achievement—while when we control for FDR, it makes substantially more discoveries and discovers 1 353 patterns. In the *Brain Cancer* dataset, for example, Spass-fdr discovers 1 471 patterns. According to a high-level analysis, the top pattern in the *Brain Cancer* dataset consists of genes involved in neural activities

$$\{\ \textit{A2BP1}, \textit{CAMK2A}, \textit{GABRA1}, \textit{GABRB2}, \textit{NRGN}, \textit{PACSIN1}, \textit{SLC12A5}, \textit{SNAP25}, \textit{SULT4A1}, \textit{SYN2}, \textit{TMEM130}, \textit{VSNL1}\ \}\ .$$

In contrast, the top pattern from the *Breast Cancer* dataset

$$\{\ \textit{AOC3}, \textit{AQP7}, \textit{BTNL9}, \textit{CIDEC}, \textit{ERG}, \textit{GYG2}, \textit{HSPB6}, \textit{ITGA7}, \textit{KCNIP2}, \textit{LPL}, \textit{PLIN1}, \textit{PPP1R1A}, \textit{SLC19A3}, \textit{TUSC5}\ \}\ ,$$

represents high co-expression of 14 membrane-related genes. We conclude that Spass manages to discover interpretable patterns also on high-dimensional real-world data.

One-Class Real-World Data

Next, we evaluate how well Spass works on unlabeled (one-group) datasets. Methods like Lamp, WYlight, or SPuManTE all require two groups, and are not applicable in this setting. We therefore compare

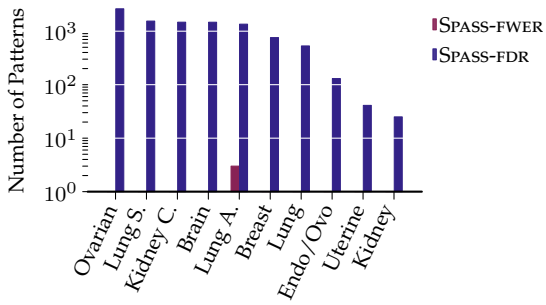


Figure 4.6: Of all competitors, only Spass-fdr can analyze high-dimensional genomics data well. We show the number of discoveries on high-dimensional, one-group and multi-group cancer genomics data from Spass-fwer and Spass-fdr only, since no competitor discovered any patterns.

to opus [190], which discovers self-sufficient patterns from data using Fisher's exact test while bounding FWER. Self-sufficient patterns are those with a frequency that is statistically significant compared to the frequencies of all subsets. opus requires the user to set a maximum number k of how many patterns it may report. As we are primarily interested in how well opus filters *redundant* patterns, we set $k = 10\,000$, which is high enough for it to discover any truly significant and non-redundant pattern.

By considering unlabeled data, we are in the application domain of pattern-set mining, which strives to discover a non-redundant set of patterns to identify informative feature co-occurrences. We compare to two state-of-the-art methods, mtv [119] and Desc [43] (cf. Chapter 2), that also rely on maximum entropy modeling.

In the left panel of Figure 4.7, we show the number of patterns discovered by mtv, Desc, opus, and Spass on 9 one-group datasets. There, we see that opus almost always reports all k patterns as significant, whereas the dedicated pattern *set* miners mtv and Desc, as well as Spass, all report similarly concise results. Closer inspection confirms that despite a rigorous FWER control, opus still returns subsets of patterns as significant discoveries. This means that testing for self-sufficiency alone is insufficient for discovering a set of non-redundant and significant patterns. We attribute this observation to the fundamental limitation of the self-sufficiency property, which tests each

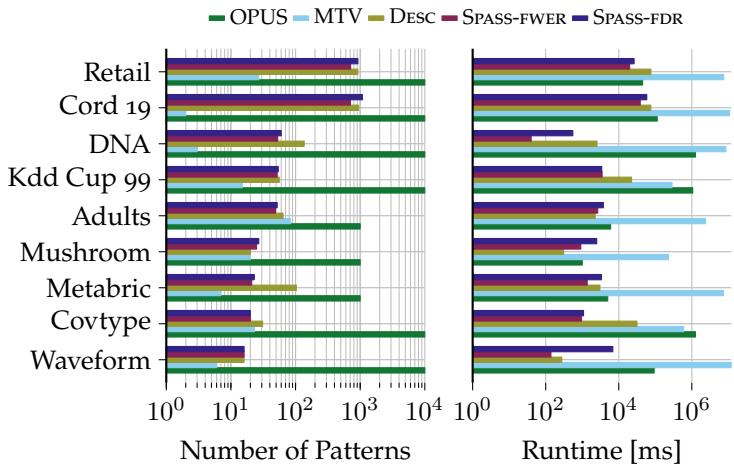


Figure 4.7: Self-sufficiency is insufficient for discovering non-redundant patterns. We show the number of statistically tested-to-be non-redundant discoveries and runtime of MTV, Desc, OPUS, and Spass on one-group datasets.

pattern *independently of prior discoveries*, and conclude that the usage of past discoveries helps to curtail redundant results.

Multi-Class Real-World Data Classification

Finally, to objectively confirm that Spass discovers characteristic and contrastive patterns that are relevant for the groups, we consider multi-group classification. Spass is neither a specialized classifier, nor do we optimize for accurate predictions. However, we can use our probabilistic model to introduce a simple Bayesian classifier

$$\hat{y}_i \leftarrow \arg\max_j \; p_i(x_i \mid S_i) \;,$$

where $\hat{y}_i$ is the likeliest prediction under Spass's distributions $p_1 \dots p_k$. While it would be too much to expect Spass to outcompete the state of the art in classification, our goal here is to objectively check if the patterns it discovers allow to separate the groups well: If they do, we will see reasonably high accuracy.

We compare Spass to decision trees as examples of nonparametric and interpretable classifiers. To train trees, we use CART [26] as a

baseline and the recent DL8.5 [4] for optimal trees. Since DL8.5 has a high computational complexity, we limit its run time to 60 minutes and its tree depth to at most 10, which is more generous than in the original study [4]. We use a 5-fold stratified cross validation to apply Spass, cart, and DL8.5 to the sampled training data (80%).

In the left panel of Figure 4.8, we report the average true positive rate on the remaining test data (20%). Even though we do not directly optimize for accuracy, we can see that Spass has a high precision in both two-group and multi-group data. Its accuracy is in most cases comparable to the results from cart. In our experiments, DL8.5 tends to have the lowest accuracy, which is especially noticeable on multi-group data.

Although we cannot directly interpret our pattern sets as trees, we can get an idea of how the model sizes compare by considering the number of rules the trees embed. To this end, we count the number of leaves—or equivalently, the number of root-to-leaf paths—in the trees, and show these together with the number of patterns that Spass discovers in the right panel of Figure 4.8. We see that cart usually requires the largest models and sometimes needs tens of thousands of rules to classify the data. In this regard, DL8.5 performs much better and often results in very small trees. However, we see that this often corresponds to a drop in accuracy of DL8.5. Additionally, we also limit tree depth of DL8.5, and therefore model size, due to its high computational complexity. cart needs much fewer rules for easy-to-classify datasets like *Metabric*. Since Spass is after descriptive models, it is no surprise that we find more significant patterns than we need to differentiate the groups. In most cases, our model is well below 100, which results in descriptions that are concise enough to be easily interpretable.

In the middle panel of Figure 4.8, we show the run time. We see that cart is usually the fastest method, closely followed by Spass. We are on median only 0.08 seconds behind cart, and in the best case, *News*, we are 6.5 minutes faster. DL8.5, on the other hand, often reaches its time limit and is slower than Spass.

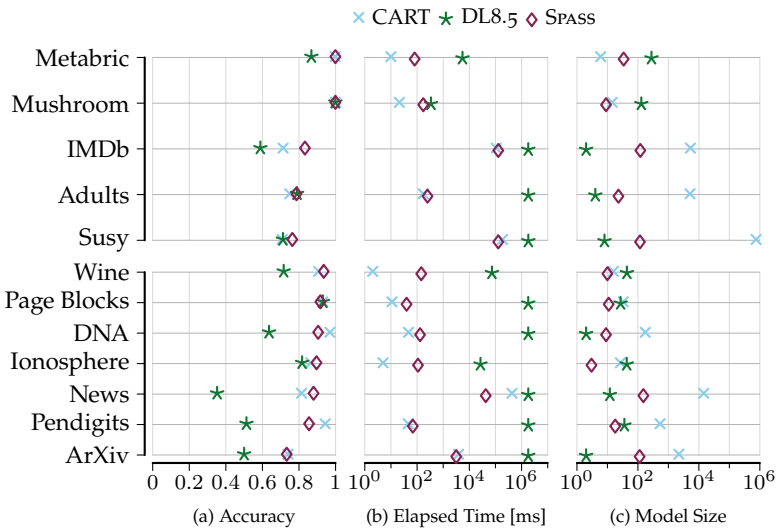


Figure 4.8: From left to right, we show the the classification accuracy (a), run time (b) and model size (c) of CART, DL8.5, and Spass as averages over a 5-fold stratified cross validation on real-world two-group and multi-group data.

Sentiment Analysis

Next, we take a closer look at the quality of the patterns that Spass discovers. To this end, we consider the IMDb movie review dataset [115], which consists of positive and negative movie reviews as text. We run Spass on this data and report associations of natural language patterns to positive or negative sentiments. After eliminating stop words, lemmatizing the corpus, and removing infrequent words, the dataset consists of 50 000 rows with 8 124 features, in which Spass discovers 215 significant patterns under FWER control, which we rank according to their significance. The top-3 patterns,

$$\{\, \textit{great fantastic} \,\},\ \{\, \textit{music sound} \,\},\ \{\, \textit{film plot twist} \,\},$$

are uniquely positive, whereas the most contrasting top-4 patterns between the two groups are

$$\{\, \textit{seen worst} \,\},\ \{\, \textit{piece crap} \,\},\ \{\, \textit{world reality} \,\},\ \{\, \textit{painful watch} \,\}.$$

Regardless of the sentiment, reviewers are concerned with *special effects*, which is the highest-ranked shared pattern.

Clinical Survival Analysis

To further analyze the interpretability of our results, we consider the problem of clinical survival analysis. In particular, we analyze the *Metabric* breast cancer dataset [40]. It consists of $2\,000$ samples (patients) over 124 binarized features, with binary labels marking the survival status of each patient. By using Spass, we discover 65 patterns at an FDR of at most 0.05. On average, these patterns consist of 4.5 ± 2.3 items, with the longest pattern having length 12. Among the patterns that are easiest to understand, we identify

$$\{\ \textit{Relapse: Recurred, Patient Died of Disease}\ \} \text{ and}$$
$$\{\ \textit{Relapse: Not Recurred, Patient Died of Other Causes}\ \}\ ,$$

as only significant for the deceased group, and

$$\{\ \textit{Survival of } 49 \textit{ Months, Relapse Free for } 31 \textit{ Months}\ \},$$

as significant for the group of survivors.

The Nottingham Prognostic Index (NPI) is an estimate of the survival chance after breast surgery, with low numbers indicating a high chance of survival. Spass discovers that a low NPI, combined with small and early-stage tumors,

$$\{\ \textit{NPI:}\ [1.0, 3.04),\ \textit{Tumor Size:}\ [1, 15),\ \textit{Tumor Stage:}\ [0, 1)\ \}\ ,$$

is associated with survivors, while high values of NPI, together with a cancerous lymph system,

$$\{\ \textit{NPI:}\ [5.06, 7.2),\ \textit{Lymph Nodes Positive:}\ [3, 45)\ \}\ ,$$

are associated with deceased patients. Significant for both groups is the association of radiotherapy and surgery type,

$$\{\ \textit{Surgery: Conserving, Radiotherapy: Yes}\ \}\ ,$$
$$\{\ \textit{Surgery: Mastectomy, Radiotherapy: No}\ \}\ ,$$

which corresponds to clinical practice.

We further discover that cancer cells which do not respond well to hormone therapy, *ER:- (by IHC)*, are typically treated with *Chemotherapy: Yes*. Spass returns two variants of this pattern: one with *Overall Survival* ≤ 49 *months* and one with *Inferred Menopausal State: Pre*, both significant for the group of deceased patients. It also discovers a pattern significant for deceased patients that characterizes the situation in which cancer cells that are hard to differentiate from regular cells (i.e., they have a high histological grade) do not respond to hormone therapy,

{ *ER:- (by IHC), ER:-, PR:-, Neoplasm Hist. Grade:* 2 } .

In these cases, hormone therapy tends to fail, surgery is very hard to perform, and hence, patients have low survival rates.

Topic Modeling

Finally, we use Spass to highlight important patterns in abstracts about different topics. To this end, we crawled ArXiv abstracts from across the machine learning topics 'explainability', 'interpretability', 'causality', 'deep learning', and some 'quantum theory' for good measure. We removed stop words, lemmatized the corpus, and removed infrequent words. Overall, the sparsely populated dataset contains 18 913 rows over 1 504 features across 5 groups, for which Spass tested 115 patterns as significant. The most characteristic pattern for 'deep learning' is unsurprisingly *deep neural network*, whereas the patterns

{ *harmonic oscillator* }, { *master equation* }

are unique to 'quantum physics'. Over all machine learning groups, the top-3 patterns are related to deep learning, namely,

{ *deep neural network* }, { *deep learning* }, { *neural network* },

which makes sense, as this is a widespread topic and the groups are fuzzy. Yet, we also discover meaningful patterns that set apart the other machine learning topics. For example, the top contrasting patterns are

{ *principle component analysis* },
{ *partial differential equation* }, { *black box* }

Finally, the most significant pattern that emerges in 'causality', 'explainability', and 'interpretability' from the 'deep learning' perspective is *structural equation*.

Overall, our experiments on real-world data from different domains therefore demonstrate that Spass not only efficiently discovers non-redundant sets of significant patterns, outperforming even specialized state-of-the-art methods, but that it also identifies meaningful patterns in practice.

4.7 Discussion

We demonstrated experimentally that Spass discovers pattern sets which are as concise as the state of the art in pattern-set mining, while retaining sequential statistical significance. Although Spass successfully yields significantly less-redundant results than the state of the art in significant pattern mining, it still leaves room for future work.

Our method is essentially a framework which permits to (i) plug in a data-appropriate probabilistic model that dependents on past discoveries; (ii) choose a statistical test; and (iii) select one of the myriad FWER or FDR control techniques [83, 84, 146, 176, 177, 188]. As such, it is easily adaptable: One can simply exchange building blocks to accommodate different types of data, such as graphs, sequences, or continuous data, or to incorporate background knowledge beyond pattern frequencies. Replacing the binomial test with the standard normal approximation, for example, yields the Z-test.

Albeit our sequential FWER control is much less conservative than Bonferroni correction, we see room for increasing the statistical power even further. Recent work, for example, introduces a novel online FWER control [177], which might yield a statistically powerful sequential FWER control. However, since this work still controls for the strict FWER, it will not replace the online FDR control, which could also be improved further. For example, we might overburden our "α-budget" by paying for each test, including tests of hypotheses that have very high p-values and thus will almost surely never result in discoveries. Therefore, we might as well outright discard (not reject) these hopeless hypotheses [176].

Further, we currently maintain *one* FDR budget for *all* groups, but it is straightforward to adapt Spass to maintain *independent* FDR budgets per group. Since we did not notice a practical performance difference in our experiments when maintaining independent budgets, we present the slightly simpler algorithm in this work.

4.8 Conclusion

We considered the problem of discovering statistically significant patterns under false discovery control. To avoid redundancy, we proposed to statistically test whether observed frequencies match with expectation, given past discoveries. To achieve high statistical power, we proposed to sequentially control for either FWER or FDR. To efficiently discover significant patterns, we introduced the Spass algorithm that uses an easy-to-compute Chernoff bound to permit efficient significance testing. Through extensive experiments, we demonstrated that our method returns concise result sets, recovers the ground truth from synthetic data, works well on data with many dimensions and any number of groups, and identifies interesting and meaningful patterns in practice. By combining pattern-set mining with significant pattern mining, Spass consistently outperforms the state of the art in both areas.

5
The Relaxed Maximum Entropy Distribution

The previous chapters are united in their use of the maximum entropy principle to discover insightful patterns. As explained in Section 2.3, this principle uniquely identifies the distribution that satisfies the constraints laid out by our model but otherwise is maximally unbiased. As soon as we consider non-trivial models, however, exact inference quickly becomes intractable. So far, we have used a static and exact factorizations of the expectation into size-constrained factors, thus limiting the expressivity of our distribution. In this chapter, we propose a relaxation that permits efficient inference by *dynamically* factorizing the joint distribution into maximum-entropy factors we can learn from data, which allows for unconstrained data modeling.

Specifically, we show that the relaxed maximum entropy (RelEnt) distribution is PAC-learnable and consistent with the standard maximum entropy (MaxEnt) distribution. Through an extensive set of experiments on synthetic and real-world data, we show that the relaxation is highly scalable, approximates standard maximum entropy very closely, allows for equally good classification as well as much faster clustering, and results in interpretable patterns.

This chapter is based on the publication: Dalleiger and Vreeken [44].

5.1 Introduction

The maximum entropy (MaxEnt) principle allows us to uniquely identify the distribution that satisfies what we know about the data, yet introduces as little other bias as possible. It is a very general principle, yet surprisingly easy to instantiate: The distribution often takes a convenient form [39], and as the resulting problem is convex, it is straightforward to optimize [45]. It is therefore no surprise that MaxEnt is useful in machine learning [104, 200], but as it provides a statistically well-founded way to measure (subjective) interestingness, it is especially useful in data mining. In data mining, the MaxEnt principle has been used to rank results given expert knowledge or beliefs [22, 87], to measure differences between data mining results [174], to identify small and non-redundant sets of informative patterns [118, 193], and to discover components in data [43].

While MaxEnt has many favorable properties, the type of knowledge we would like to incorporate into our model strongly affects how efficiently we can infer it. As long as we care only about overall densities, life is simple [22, 174]. But whenever we incorporate dependencies between attributes, such as basic co-occurrence frequencies (e.g., a and b co-occur in 50% of the data) [43, 118], it becomes PP-hard to infer the resulting model [172]. Hence, whenever we want to use MaxEnt with non-trivial factor models, we need to employ tricks to achieve a reasonable runtime.

The main trick in the literature is to construct not one maximum entropy distribution p^* over all attributes, but rather factorize the distribution according to the independences in our model [43, 118, 193]. That is, if those are given as a set of statistics S, we can partition this set into independent subsets S^i, and we can later infer the maximum entropy distribution p^* for each S^i. This, of course, only works when S indeed consists of independent parts. Rather than hope for the best, existing methods hence enforce this—for example, by allowing only up to k mutually dependent statistics in S, or by partitioning the attributes and disallowing any statement about two or more parts. Putting such restrictions on S obviously strongly limits the expressivity of the result-

ing models, and as our experiments confirm, it leads to underfitting the data.

In this chapter, we take a different approach, starting from the observation that not everything we know is always equally relevant. That is, we propose to relax the maximum entropy distribution. Instead of using every piece of our model $s \in S$ for every inference, we suggest to leverage only the *most relevant* subsets of S, working with *different* maximum entropy distributions *depending* on what we infer for. In other words, rather than enforcing one static factorization for all queries, we consider different, dynamic factorizations of S depending on the inference.

We prove that our proposal, which we refer to as the *relaxed maximum entropy distribution*, is both PAC-learnable and consistent with vanilla maximum entropy. Furthermore, we elucidate how the dynamic factorization problem relates to data summarization, which allows us to specify an extremely fast instantiation based on pattern mining. We show that our approach, which we call Reap, allows us to consider almost arbitrarily large models and approximate complex ground-truth distributions *better* than the strongly constrained existing solutions, while at the same time being many orders of magnitude faster. Moreover, through extensive experiments on both synthetic and real-world datasets, we show that our relaxed distribution outperforms vanilla maximum entropy at multiclass classification, pattern set mining, and data decomposition, in terms of both quality and speed.

In sum, our main contributions are that we

1. introduce the relaxed maximum entropy distribution,
2. provide a practical realization of this distribution,
3. show how factors of the model relate to associations in the data,
4. provide a highly efficient algorithm to estimate the distribution through pattern discovery, and
5. demonstrate, through an extensive set of experiments, that the distribution and the algorithm work well in practice.

The remainder of this chapter is structured as follows. Having introduced the maximum entropy distribution already in Section 2.3, we develop the theory of our relaxed maximum entropy distribution in

Section 5.2, propose our Reap algorithm to find a good relaxed maximum entropy model efficiently in Section 5.3, and contextualize related work in Section 5.4. In Section 5.5, we demonstrate that Reap works well in practice, before concluding in Section 5.6.

5.2 Theory

In this section, we briefly introduce our notation, and we develop the theory of our relaxed maximum entropy distribution.

5.2.1 Notation

In this chapter, we also consider binary tabular data, and largely follow the notation from Section 2.2 and Section 4.2. In a nutshell, for any finite sets A and B, we denote 2^A for the powerset of A, $\binom{A}{k}$ for the set of all subsets of A of size $k \in \mathbb{N}$, $A \bigtriangleup B$ for the symmetric difference between A and B, and $[n] = \{1, 2, \ldots, n\}$ for any $n \in \mathbb{N}$. The indicator function is $\mathbf{1}$, all logarithms are to base 2, and by convention, we use $0 \log 0 = 0$.

As we again consider binary tabular data, a dataset X is a multiset of n samples from the set $\Omega = 2^{\mathcal{I}}$ of all possible samples, over d features in the set $\mathcal{I}$. Like in Section 2.2, for a given partitioning of $\Pi \in \varpi(X)$, we denote the $k \geq 1$ groups in Π by $\{X_1, \ldots, X_k\}$ and let $n_i = |X_i|$. Our method requires an underlying probabilistic model, for which we again choose the maximum entropy distribution introduced in Chapter 2.

5.2.2 Relaxation

Building on our introduction of the original maximum entropy distribution in Section 2.3, we now develop our relaxation of this distribution. To this end, we first recapitulate the factorization of our expected frequency in more detail. We then formalize the *static factorization* and contrast it with our *dynamic relaxation*. In Section 2.3, we established that a straightforward inference of the expectation p of our maximum entropy distribution involves an exponential number of terms in the

sum. Many of these terms evaluate to the equivalent probabilities, and they can be used to partition Ω into equivalence classes $\Omega_{/\sim}$, with

$$x \sim y \iff f(x \mid S) = f(y \mid S)$$

for $x, y \in \Omega$, such that the expectation becomes the weighted sum

$$\mathbb{E}_f[x \mid S] = \sum_{\substack{[\![y]\!] \in \Omega_{/\sim} \\ x \subseteq y}} |[\![y]\!]| \, f(y \mid S)$$

over these groups. Mampaey, Vreeken, and Tatti [118] showed how to create the set of equivalence classes

$$\{[\![y]\!] \in \Omega_{/\sim} \mid x \subseteq y\}$$

that support x and their weights from S, such that the size scales exponentially only in S instead of $\mathcal{I}$. However, if S is sufficiently large, the inference is still intractable. Hence, the question arises if we can reduce the inference complexity without reducing the size of S.

If there exists a valid factorization of p into independent factors $\prod p_i$, we will not change the outcome by inferring the factors independently from one another. Conversely, we will not lose information by factorizing S into subsets S^i that are independent in p. Whenever we can do so, the inference complexity of each factor $p_i(\cdot \mid S^i)$ scales only in S^i, and if the sizes of these subsets S^i of S are now considerably smaller than S, we achieve a significant gain in inference complexity without loss.

Example 5.1. Consider the case of

$$S = \{abc, cd, de, df, ef\},$$

where the set $\mathcal{I}$ consists of letters from a to f. If we know that the letters abc are independent of the rest, we can factorize S into $S^1 = \{abc\}$ and $S^2 = \{de, ef, df\}$ without information loss. The inference of the frequency of the pair ab only marginalizes out the c of factor S^1. However, if S^1 and S^2 are *not* independent (for example, if the pair cd is

part of our statistics), then we have to marginalize out c, cd, de, ef, df, which consists of the sum over 2^5 combinations.

More formally, the inference of the expectation $p(x \mid S)$ becomes the product

$$\prod_{S^i} p(x \cap s^i \mid S^i)$$

of individual maximum entropy factors, where the set $s^i = \cup S^i$ contains the elements that are *associated* through S^i. In the following, we generalize this observation in terms of a factorization oracle φ.

Definition 5.1 (Generalized Factorization). For a *given* factorization oracle $\varphi \in \Omega \to 2^S$ that is provided with statistic $S \subseteq \Omega$, the *generalized factorization* is

$$\tilde{p}(x \mid S) = \prod_{S^i \in \varphi(x)} p(x \cap s^i \mid S^i) \,, \tag{5.2.1}$$

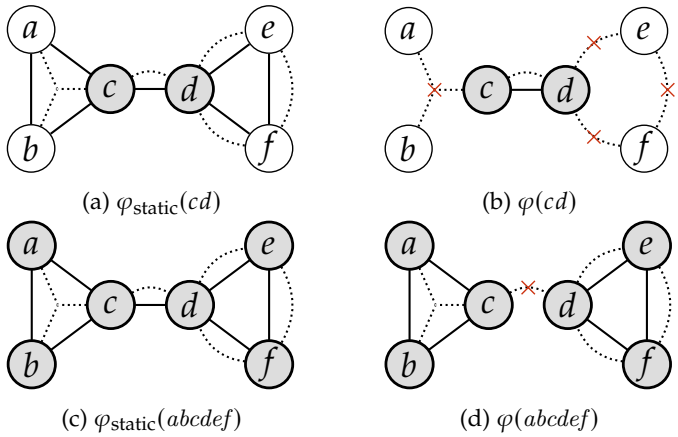
where the factors $p(\,\cdot \mid S^i)$ have maximum entropy, subject to constraints imposed by S^i (Eq. (2.3.1) and (2.3.2)).

Example 5.2 (Static Factorization). Assume that our statistic S and the factorization of p are given and fixed. This means that we have access to the set of p-independent statistics $\{S^i\}_i$ such that $S = \bigsqcup_i S^i$. In this situation, the *static factorizer* is

$$\varphi_{\text{static}}(x) = \left\{ S^i \in \mathcal{S} \middle| x \cap s^i \neq \emptyset \right\} \,.$$

In theory, if $\mathcal{S}$ truly models the independences of the ground-truth distribution, using φ_{static} is optimal. In practice, however, modeling the true factorization can pose a significant problem: The complexity of inferring a single factor is still exponential in the size of S^i. To circumvent this issue, we have to drastically limit the size of each S^i to be no greater than, say, a user-defined $\beta \in \mathbb{N}$. This limited special case of Eq. (5.2.1) plays a central role in the inference of the maximum entropy models as used by, e.g., MTV [118] and Desc [43] (cf. Chapter 2).

The problem is that if we use the *static* factorization, we have to choose between tractable inference complexity and sufficiently rich modeling of the data. For example, consider the static factorization



(a) $\varphi_{\text{static}}(cd)$ (b) $\varphi(cd)$

(c) $\varphi_{\text{static}}(abcdef)$ (d) $\varphi(abcdef)$

FIGURE 5.1: We depict two exemplary factorizations as graphical models. For $S = \{abc, cd, de, df, ef\}$, denoted by the dotted edges, we visualize possible factorizations for queries cd (top, a–b) and $abcdef$ (bottom, c–d). A solid edge means that the corresponding nodes partake in the inference. For example, as $\{abc\} \in S$, the *static factorization* has to marginalize-out a and b, in order to infer cd. Depicted by the presence of solid edges (left), we see that the static factorization requires all of S. Right, the *dynamic factorization*, however, efficiently selects the cd factor, thus crossing-out dependencies. Likewise for $abcdef$, we see that the static factorization (left) again uses the complete graph, whereas the dynamic factorization (right) frugally factorizes into two easy-to-infer cliques, thus trading inference complexity with information loss.

$\mathcal{S}$ that consists of the two independent factors $\{abc\}$ and $\{de, df, ef\}$. If we model an association between c and d, we introduce a statistical dependency between the two factors, and hence, $\mathcal{S}$ becomes $\{\{abc, cd, de, df, ef\}\}$. However, as the size of this factor exceeds the budget of $\beta = 4$, we are prohibited from modeling dependency cd. In the following, we introduce a relaxed, more flexible factorizer that relieves us from this choice.

EXAMPLE 5.3. In Figure 5.1, we consider a graphical representation of $\tilde{p}$ for our example set $S = \{abc, cd, de, df, ef\}$, which we visualize using dotted edges. We show possible factorizations of the two queries $abcdef$ and cd, using both static factorization (Figure 5.1c resp. 5.1a) and relaxed factorization (Figure 5.1d resp. 5.1b). The static factorization involves the complete graph, and therefore, the graph is connected (solid lines) and the inference is quite complex. However, if we deliberately

ignore dependencies, we can partition the graph into clique graphs that are less complex to infer by cutting out edges (the crossed-out dotted edges).

In other words, by essentially cutting out edges from the graph, we can, at the cost of information loss, tremendously reduce the inference complexity. In general, the inference complexity of a factorization is

$$\gamma(\mathcal{S}) = \sum_{S^i \in \mathcal{S}} 2^{|S^i|} ,$$

and for a given factorization oracle φ that deliberately ignores associations, the reduction in inference complexity,

$$\gamma(\mathcal{S}^*) - \gamma(\varphi(x)) \in O(2^\delta) ,$$

is *exponential* in δ, where δ is the difference between the sizes of the largest factors in $\mathcal{S}^*$ and $\varphi(x)$. For example, the complexity in Figure 5.1a is $\gamma(\{S\}) = 2^5$, whereas by omitting cd in Figure 5.1a, the complexity drops to only $\gamma(\{\{abc\}, \{de, df, ef\}\}) = 2^1 + 2^3$.

Dynamic Factorization In the example above, we *dynamically* adapt the factorization of $\tilde{p}$ to trade inference complexity with information loss for the queries $x \in \Omega$. To formalize this idea, we assume that not all information in statistics S is necessarily worth including in the factorization of each $x \in \Omega$. More precisely, we assume that there is a subset of S that contributes very little to no information to the expected frequency of x. Similar to the example, if we avoid maintaining these statistics in our factorization, we can reduce the inference complexity exponentially while keeping information loss at a minimum. Put formally, we want the factorization of $\tilde{p}$ that loses the least amount of information while being tractable to infer, i.e.,

$$\varphi_{\text{dynamic}}(x) = \underset{\mathcal{S} \subseteq 2^S}{\arg\min}\ D(p^* \| \tilde{p}_{\mathcal{S}}) \text{ s.t. } \gamma(\mathcal{S}) < \beta , \tag{5.2.2}$$

for a $\beta \in \mathbb{N}$, where D is the Kullback-Leibler divergence (Eq. (2.5.2)).

However, directly solving Eq. (5.2.2) has three drawbacks:

1. The total number of possible factorizations of $\tilde{p}$ is exponential in the size of S.
2. Each of these factors is supposed to maximize its entropy.
3. Computing D is computationally costly.

Together, this all has to be done *at least* for every $x \in X$, and hence, this is not a very practical factorizer. In the following, we introduce an alternative that does not suffer from the drawbacks just sketched (i–iii).

For correctness, we require that our φ splits any x into pairwise disjoint factors. In other words, we seek a factorization $\varphi(x)$ in which any two factors S^i and S^j never cover the same subset of x, i.e.,

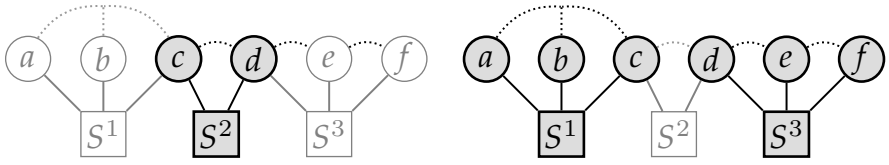
$$x = \bigsqcup_{S^i \in \varphi(x)} x \cap s^i \qquad \forall x \in \Omega \,.$$

For efficiency, we require that each factor generated by φ must be efficiently inferable, say,

$$\varphi \in \mathcal{O}(\mathrm{poly}(|\mathcal{S}|)) \,,$$

for some set $\mathcal{S}$ (i, iii). This also means that we need efficient (i.e., poly-time (ii)) access to all factors. Since maximizing the entropy of a single factor has a complexity that is exponential in S^i, we cannot simply generate arbitrary factors for any inference. Having to choose maximum entropy factors from a list, however, meets this demand. Hence, our factorizer selects factors $S^i \in \varphi(x)$ from a set of pre-determined maximum entropy factors known to the factorizer beforehand, which we call *elementary factors*. As we only require that the factorization is correct, it becomes unnecessary that elementary factors are disjoint, which has a higher modeling capability than the static factorization. That is, we *relax* constraints and are still correct, which provides additional flexibility used to efficiently model dependencies hat have previously been impossible to model efficiently by φ_{static}, due to budget constraints. Formally speaking,

Definition 5.2 (Practical Factorizer). Let $\varphi \in \Omega \to 2^{\mathcal{S}}$ be a factorizer for the efficiently computable cost function of a factor $c \in \Omega \to \mathbb{R}$. For



(a) $\varphi(cd) = \{S^2\}$ (b) $\varphi(abcdef) = \{S^1, S^3\}$

Figure 5.2: We visualize our example as a factor graph. For the given elementary factors $\mathcal{S} = \{\{abc\}, \{cd\}, \{def\}\}$, we show factor graph representations of $\tilde{p}$ for inferring cd (left) and $abcdef$ (right). We represent each elementary factor $S^i \in \mathcal{S}$ as a square and highlight the factors that are in $\varphi(x)$.

any given set $\mathcal{S}$ of elementary factors, and for each $x \in \Omega$, our factorizer $\varphi(x)$ finds factorizations by

$$\varphi(x) = \operatorname*{arg\,min}_{A \subseteq \mathcal{S}} \; c(A) \;\; \text{s.t.} \;\; \bigsqcup_{A^i \in A} x \cap a^i = x \,.$$

In all its generality, this definition of a practical factorizer allows for many efficient, correct, *however* different factorizations of each element x. To algorithmically identify one, we need to specialize this practical factorizer further—by making a choice. Since we ultimately infer the expected frequency of patterns, we interpret the coverage (by φ) as an explanation for its estimate, revealing precisely which feature dependencies are used and which are omitted.

Example 5.4. Assume that in addition to S from our running example, we are now additionally given a set $\mathcal{S}$ of elementary factors $\{\{abc\}, \{cd\}, \{def\}\}$. In Figure 5.2, we depict factor graph representations of $\tilde{p}$ and visualize possible factorizations of two different queries. Again, we trade inference complexity with information loss, however this time, we do so not by removing arbitrary edges from the graph but by selecting a subset of elementary factors (squares) from $\mathcal{S}$. In this figure, we also illustrate that elementary factors are *not* necessarily disjoint. The factors in $\varphi(x)$, however, are always disjoint for any x.

Following Occam's Razor, we seek to provide the simplest factorization. That is, we are specifically interested in providing the smallest number of factors that cover $x \in \Omega$. We call this factorization the *minimal sufficient factorization* of x. To do this, we simply parameterize

our practical factorization by letting the cost function of a factor be $c(A) = |A|$. By doing so, the practical factorization problem becomes, in fact, a variant of the *minimal exact set cover* or hitting set problem [93], and is therefore efficiently optimizable by taking a greedy approach. In other words, to overcome issues (i) and (iii), we take a greedy approach to the set cover problem, with a complexity of $O(k^2)$. Overall, the worst-case complexity of inferring the relaxed expectation,

$$O(2^\beta k^2) \ ,$$

is bounded by a term that is exponential in the maximally allowed size of a factor β. The expected average complexity X is less than this, since we usually do not exhaust the full budget β and efficiently stop greedy set cover early.

To summarize the above, we have introduced an efficiently computable φ that factorizes $\tilde{p}$ by selecting the minimal sufficient explanation in terms of given, pre-determined, not necessarily disjoint maximum-entropy factors. Provided with a set of elementary maximum-entropy factors, we next use this factorization to *relax* the inference of our expectation p^*. Afterwards, we describe how to obtain elementary factors from data, which we summarize as our algorithm.

Relaxed Maximum Entropy model Estimator We want to find the set of elementary factors $\mathcal{S} \subseteq 2^\Omega$ that minimizes the divergence between the true reference distribution p^* and our relaxation $\tilde{p}$ such that the inference complexity of $\tilde{p}$ is bounded. Formally:

Problem 5.1 (Relaxed maximum entropy). Given a *budget* $\beta \in \mathbb{N}$, we seek to

$$\begin{aligned} \underset{\mathcal{S} \subseteq 2^\Omega}{\text{minimize}} \quad & D(p^* \| \tilde{p}_\mathcal{S}) && (5.2.3) \\ \text{subject to} \quad & \left|S^i\right| \leq \beta && \forall S^i \in \mathcal{S} \\ & f_{S^i} \leftarrow \text{Eq. (2.3.1) w.r.t. } S^i && \forall S^i \in \mathcal{S} \\ & \tilde{p}(x) = q(x) && \forall x \in \cup\mathcal{S} \ , \end{aligned}$$

where D is the Kullback-Leibler divergence between the given target reference distribution p^* and the relaxation $\tilde{p}$. Every f_{S^i} maximizes the marginal entropy according to Eq. (2.3.1). From a set of equally diverging relaxations, we select the one with the lowest inference complexity.

Even though this problem is straightforward to write down, it is challenging to solve, as the search space of size 2^{2^Ω} is tremendously large. We also cannot access the true factorization $\mathcal{S}^*$ to guide our search for the optimal solution. Furthermore, we cannot actually infer the unknown distribution p^*. Hence, we cannot simply solve Problem 5.1 by means of an off-the-shelf combinatorial optimization algorithm. Therefore, in the following, we show that Problem 5.1 is learnable (i), by discovering associations (ii), from data (iii).

We start with the mild assumption that the support of $\tilde{p}$ subsumes the support of p^*, i.e.,

$$\operatorname{supp} p^* \subseteq \operatorname{supp} \tilde{p} \ ,$$

from which follows that the divergence $D(p^* \| \tilde{p}) < \infty$ is finite [95], and therefore, that a solution must exist. With the following lemma, we show that there actually exists a solution that approximates p^*.

Lemma 5.1 (Problem 5.1 is PAC-learnable). The probability that the divergence is sufficiently small converges to 1, i.e.,

$$\mathbb{P}\left[D(p^* \| \tilde{p}) < \epsilon\right] \to 1 \ ,$$

where $\epsilon \to 0$ for $|\mathcal{S}| \to |\mathcal{S}^*|$.

Proof. Let $Y \sim p^*$ and $X \sim \tilde{p}$ be two random variables. We know that if the conditional entropy $H(Y \mid X)$ is 0, the divergence $D(p^* \| \tilde{p})$ is minimal. Furthermore, we know from Fano's inequality [37] that the conditional entropy $H(Y \mid X)$ of $Y|X$ is bounded from above by the error probability $\mathbb{P}(E)$ for the random variable $E = (X \neq Y)$, in the sense that

$$H(Y \mid X) / \log |\Omega| < \mathbb{P}(E) \ .$$

This means that as long as $\mathbb{P}(E)$ converges to 0, the conditional entropy converges to 0. By creating an elementary factor $S^y = \{y\}$ for such a $Y = y$, we create a maximum entropy factor $p_y \in \mathcal{P}_{S^y}$ for which $p(y) = p^*(y)$ is true by construction, and hence, if we make use of S^y, we reduce the probability of an error. Thus, the set $\mathcal{S} = \{\{x\}_{x \in S^*}\}$ is an example of a sequence of factors for which $\mathbb{P}(E)$ converges asymptotically, given that the moment constraints are consistent. Hence, our problem is learnable according to Lemma 5.1. □

Now we know that the problem is learnable in general, but the distribution p^* is still unknown. Next, we would like to get rid of this unknown, and we seek to specify the empirical information loss caused by the factorization. To these ends, we need the following lemma.

Lemma 5.2 (Asymptotically Consistent). For a given set of n samples $X = \{x_i\}_{i \in [n]}$ from p^*, where $X \sim p^*$, the empirical estimator $\widehat{D}^n$ of Eq. (5.1),

$$\lim_{n \to \infty} \widehat{D}^n(p^* \| \tilde{p}) \to D(p^* \| \tilde{p}) \,,$$

converges asymptotically to D.

Proof. We assume $\operatorname{supp} q \subseteq \operatorname{supp} p^*$. We write $D(p^* \| \tilde{p}) = D(p^* \| q) + D(q \| \tilde{p})$ by using the information projection [39]. Since $X \sim p^*$, and due to the law of large numbers, we know that $\lim_{n \to \infty} \widehat{D}^n(p^* \| q) \to 0$. Thus, it is sufficient to show

$$\lim_{n \to \infty} \widehat{D}^n(q \| \tilde{p}) \to D(q \| \tilde{p}) \,,$$

which is trivially true because q is the empirical estimate. □

With Lemma 5.2, we can now state the empirical information loss introduced by our factorization for a single $x \in \Omega$ as

$$\log q(x) - \sum_{S^i \in \varphi(x)} \log p(x \cap s^i \mid S^i) \,. \tag{5.2.4}$$

Next, we focus on how to find elementary factors. Even when using the empirical estimator, solving the problem directly would involve

search in the very large space of 2^{2^Ω} possible combinations of elementary factors. To overcome this obstacle, we introduce a way to considerably reduce the search space *without* loss. For this, we show that we can limit our search to a set of elements $S \subseteq \Omega$ that are statistically dependent in data X. To do so, we first formalize what we mean by dependencies. We say that $x \in \Omega$ is *conditionally independent* of $y \in \Omega$ if

$$x \perp\!\!\!\perp y \mid \mathcal{S}^* \iff \nexists S^j \in \mathcal{S}^* : x \subseteq s^j \wedge y \subseteq s^j \ .$$

By definition, there is no single maximum-entropy factor S^i in the assumed-to-be given true factorization $\mathcal{S}^*$ that contains both $x \in \Omega$ and $y \in \Omega$ that are statistically independent in X. Therefore, we have the following lemma.

Lemma 5.3 (Factors from Associations). For a given set $S^* \subseteq \Omega$ that contains all statistically dependent sets of elements $x \in \Omega$, there are no factors S^i in $\mathcal{S}^*$ that contain $x \notin S^*$.

Proof. Assume otherwise, that is, assume that there exists $x = a \cup b \in S^*$ for which $a \perp\!\!\!\perp b \mid \mathcal{S}^*$. Thus, $\nexists S^{ab} \in \mathcal{S}^*$ with $a, b \in S^{ab}$. Hence,

$$p^*(a \cup b) = p^*(a \mid S^a) p^*(b \mid S^b) \ .$$

However, by definition, $q(a \cup b) \neq q(a)q(b)$ holds true. The contradiction then follows from $\lim_{n\to\infty} q^{(n)} \to p^*$. □

In other words, instead of minimizing Eq. (5.2.3) directly, firstly, it *suffices* to construct a factorization $\mathcal{S}$ that minimizes divergence, for a given to-be-found set S (Lemmas 5.1 and 5.3). Secondly, it *suffices* to discover S from X (Lemma 5.2). We separate these two problems and solve them in turn. To this end, we start by creating the set of elementary factors $\mathcal{S}$ from a given set of dependencies S. Next, we discover this $S \subseteq \Omega$ from data X.

We start with a given set S. If S is provided and fixed, Problem 5.1 *simplifies* to the task of selecting a factorization

$$\underset{\mathcal{S} \subseteq 2^S}{\text{minimize}} \ D(q \| \tilde{p}) \text{ s.t. constraints from Eq. (5.2.3) are fulfilled} \quad (5.2.5)$$

from the powerset of S. On the other hand, if the factorization function is known, our problem *simplifies* to discovering the set S, allowing us to rephrase Eq. (5.2.3) in terms of minimizing the cross-entropy between p^* and $\tilde{p}$,

$$H(p^* \| \tilde{p}) = -\sum_{x \in \Omega} p^*(x) \log \tilde{p}(x) \ .$$

As a direct consequence of Lemma 5.2, we use the empirical cross-entropy $\hat{H}$, from which we derive the following problem.

Problem 5.2 (Summary Problem). For a given factorizer φ, we call the problem of selecting the set $S \subseteq \Omega$ with the highest regularized likelihood,

$$\underset{S \subseteq \Omega}{\text{minimize}} \quad \ell(\hat{\mathcal{S}}) = -\sum_{x \in X} \log \tilde{p}(x \mid \hat{\mathcal{S}}) + r(\hat{\mathcal{S}}) \ ,$$

the *summary problem*, where $\hat{\mathcal{S}}$ is a solution to Eq. (5.2.5) for S. To prevent overfitting and limit the total number of constraints, we use the Bayesian Information Criterion (BIC)

$$r(\hat{\mathcal{S}}) = \frac{1}{2} \log |X| \sum_{S^i \in \hat{\mathcal{S}}} \left| S^i \right| \ ,$$

where the number of degrees of freedom in our model is the combined size of all elementary factors.

5.3 Algorithm

In the following, we derive an algorithm that discovers a relaxed maximum entropy model $\tilde{p}$ from data. As described in Section 5.2, we split this task into two: firstly, discovering patterns S in X, and secondly, estimating elementary factors $\mathcal{S}$ for S. However, there are 2^{2^I} possible sets S to select from, and many more elementary factors $\mathcal{S}$. Therefore, we resort to an iterative approach that jointly discovers associations and elementary factors, whose outline follows next.

Given a set S and fixed elementary factors $\mathcal{S}$, we iteratively obtain the candidate $x \in C \subseteq \Omega$ with the highest estimated gain using heuristic $h \in \Omega \to \mathbb{R}$, for which we then create new factors. That is, starting

with no dependencies $\mathcal{S} = \{\{x\} \mid x \in S\}$ between all items $S = \mathcal{I}$, we iterate

$$S \leftarrow S \cup \arg\max_{x \in C} \; h(x) \,, \tag{5.3.1}$$

and generate new elementary factors of size at most β, until convergence of ℓ. Next, we explain these steps in detail, starting with the creation of new elementary factors from candidates, followed by the heuristic h.

Creating Elementary Factors from Candidates We describe the creation of elementary factors for a given pattern candidate $x \in \Omega \setminus S$. As mentioned in Eq. (5.2.5), our goal is to minimize the divergence between our relaxed maximum entropy distribution model and the empirically observed frequency distribution. To efficiently capture the dependencies modeled by x, we need one corresponding elementary factor that contains all observed information about x. If there is such a factor $S^x \in \mathcal{S}$, we require

$$q(x) = p(x \cap s_x \mid S^x) \,,$$

or otherwise, we create a new factor $S^x \subseteq S$ with access to necessary information from S and q about x.

Because our factorizer φ selects the information about x, we incorporate its explanation in our newly created factor. Thus, to minimize the divergence, we include our past explanation for x and x itself

$$\mathcal{S} \leftarrow \mathcal{S} \cup \{\varphi(x) \cup \{x\}\} \,. \tag{5.3.2}$$

into a new factor. However, as new elementary factors can easily exceed our budget β, we need to limit the factor size. To achieve this, we use the least diverging factor that is under budget, by solving the constrained problem below.

Problem 5.3 (Relaxing a Factor). For a given factorizer φ, and provided with a set of elementary factors $\mathcal{S}$, relaxing any maximum entropy factor $p(\,\cdot \mid S^i)$ for $S^i \in \mathcal{S}$ is the problem of selecting the subset

of at most $\beta \in \mathbb{N}$ elements with the smallest information loss, i.e.,

$$S^i \leftarrow \underset{A \subseteq S^i, |A| \leq \beta}{\arg\min} D_{\Omega^i}(q \| p_A)\,, \tag{5.3.3}$$

where D_{Ω^i} is the Kullback-Leibler divergence with respect to the partition $\Omega^i \subseteq \Omega$ that factor S^i supports.

Fortunately, β tends to be small in practice, and the divergence D is known to be submodular (cf. Lemma 2.1) [43, Proof in App. 1]. Therefore, we can solve Eq. (5.3.3) greedily with guarantees. To specifically relax S^x, we ensure that x is always present in that factor after the relaxation.

Running the iterative method outlined in Eq. (5.3.1), however, might create factors that are superseded by other factors at a later iteration, thus do not participate in any factorization of X—and consequently are not needed. Formally, elementary factors $A \in \mathcal{S}$ are *unused* iff $\nexists x \in X$ such that $A \in \varphi(x)$. This allows us to easily identify unused elementary factors after the final iteration, remove them from the model, and reduce our model selection penalty $r(\mathcal{S})$.

Discovering the Statistics Now that we know how to iteratively create new elementary factors, we can explain how we discover S. To do so, we will specify the heuristic h that we use to rank candidates from set C. Afterwards, we show a simple way to build up C. The problem is that C becomes quite large, and ℓ is expensive to compute. Hence, we do not compute the ℓ to rank each element in C exactly. Instead, we make use of a cheaper-to-compute heuristic h that is based on the pointwise loss caused by the factorizer. This pointwise score from Eq. (5.2.4) evaluates the loss as if we only made use of the candidate once. However, the usage for each resulting candidate factor can differ greatly. We thus discount the loss using the candidate-factor usage

$$u(x) = |\{A \in \varphi(y \mid \mathcal{S}') | y \in X\,, x \subseteq \cup A\}|\ ,$$

where $\mathcal{S}' = \mathcal{S} \cup \{\varphi(x) \cup \{x\}\}$ (according to Eq. (5.3.2)). By scaling Eq. (5.2.4) with $u(x)$, we obtain our heuristic

$$h(x) = u(x) \left[\log q(x) - \textstyle\sum_{S^i \in \varphi(x)} \log p_{S^i}(x) \right] .$$

This leaves us to specify the candidate set C. Naïvely, we could set $C = \Omega$. However, this is not practical. Ω is typically prohibitively large, and it contains exponentially many candidates that will be uninformative with regard to S. We hence propose a more effective breadth-first search strategy, in which we take into account what S can already explain well. In a nutshell, we iteratively generate candidates by merging pairs $x, y \in S \cup \mathcal{I}$ into a candidate $x \cup y \in C$. From all the candidates in C, we are only interested in the candidates from which we expect a reduction of the BIC score ℓ. If $h(z)$ is less than the actual cost of inserting z in terms of our regularizer r, we remove z from C. The actual cost is $r(z) = r(\mathcal{S} \cup \{S^z\}) - r(\mathcal{S})$, and it is not necessarily equal for all candidates since it depends on the factorization of z.

Putting the above together, we have Reap, the Relaxed maximum Entropy Accelerated Pattern-set miner, whose pseudocode we give as Algorithm 5.1. In short, starting with the singleton-only model (lines 1–2), we generate our initial batch of candidates C (line 3). We consider these candidates in descending order of h (line 5), and evaluate the best candidate $x \in C$ (lines 5–8). If the objective function improves, we keep the candidate and the factorization (lines 9–11), otherwise, we reject it. Lastly, we remove unused factors (line 12).

The computational complexity of Reap depends on the number of candidates in C, which is quadratic in the size of S, and in the worst case, can grow up to $|\Omega|$. Hence, it is in $O(2^d)$, where d is the number of features.

5.4 Related Work

The principle of maximum entropy was proposed by Jaynes [85, 86] as a general approach to choosing probability distributions. The theoretical foundations were further developed by, among others, Csiszár [39], who showed that the maximum entropy distribution minimizes the

Algorithm 5.1: **Reap**

Input: Data X, Factorizer φ
Output: Factors $\mathcal{S}$, Pattern-set S

$S \leftarrow \mathcal{I}$
$\mathcal{S} \leftarrow \{\{x\} \mid x \in \mathcal{I}\}$
$C \leftarrow \{z = x \cup y \mid x, y \in S, r(z) < h(z)\}$
while $C \neq \emptyset$
    $x \leftarrow \arg\max_{z \in C \setminus S} \; h(z) - r(z)$
    $C \leftarrow C \setminus x$
    $S^x \leftarrow$ minimally divergent factor with Eqs. (5.3.2) and (5.3.3)
    if $\ell(\mathcal{S} \cup \{S^x\}) < \ell(\mathcal{S})$
        $S \leftarrow S \cup \{x\}$
        $\mathcal{S} \leftarrow \mathcal{S} \cup \{S^x\}$
        $C \leftarrow \{z = x \cup y \mid x, y \in S, r(z) < h(z)\}$
remove unused factors from $\mathcal{S}$ and let $S \leftarrow \bigcup \mathcal{S}$
return $(\mathcal{S}, S)$

Kullback-Leibler divergence to the uniform distribution, that it has an exponential form, and that its maximization is convex. For large sample spaces Ω, the main bottleneck is the computation of expectations. Tatti [172] showed that for the case of itemset frequencies, this computation is PP-hard.

We do not always have to compute the distribution, as we can also approximate it. Barron and Sheu [15] show that under moment constraints, this is possible in terms of exponential families and basis function expansion using, e.g., polynomials. Bierig and Chernov [23] studied how Monte Carlo methods of approximate and exact inference can be used to approximate the maximum entropy distribution. Singh and Vishnoi [163] establish the equivalence of maximum entropy inference to general counting problems, and show that we can use approximate counting techniques to approximate the distribution. Approximate counts immediately translate to noisy, and therewith relaxed, moment constraints. Dudík et al. [50] present a maximum entropy problem with relaxed constraints that are generalized regularization measures in their dual form. For possibly noisy generalized

constraints, Sutter et al. [170] propose an approximation strategy for the dual of the maximum entropy problem by means of a fast gradient approximation.

To the best of our knowledge, probabilistic independences have first been exploited for the factorization of the maximum entropy distribution into efficiently inferable factors by Mampaey et al. [118]. However, to enforce the efficient inference in practice, the authors had to additionally constrain this factorization. Despite being constrained, this factorization has been successfully used to discover concise and non-redundant pattern sets [118], sample realistic categorical datasets [193], and discover pattern compositions [43] (cf. Chapter 2).

Our method Reap builds on probabilistic dependencies in data, which we ascertain using information theoretical principles. In practice, we achieve an approximation by minimizing the information divergence via the search for probabilistic dependencies in data, i.e., patterns. As pattern mining aims to discover co-occurring items in data that are, e.g., frequent (i.e., [5, 133]), informative (cf. Chapter 2), or statistically significantly correlated (cf. Chapter 4 and Chapter 3), we are closest to informative pattern-set mining, such as Desc [43] or mtv [118]. Not only does this mean that Reap is closely related to pattern mining, we also explain how the structure of a (maximum entropy) probabilistic graphical factor model is linked to informative pattern-set mining. Where Reap, however, uses informative patterns to balance the conciseness of its factorization with the conciseness of each factor, pattern-set mining is after concise and informative pattern-sets. Going beyond informativeness, novel approaches to pattern-set mining (as we have deployed in Chapter 4) lift the dependency assessment from being probabilistic to statistical, to increase robustness against noise, trustworthiness, and interpretability by domain experts. Although useful for mining patterns, we are after approximating the maximum entropy distribution by minimizing the information divergence directly.

5.5 Experiments

In our experiments, we evaluate Reap on synthetic data as well as on 57 real-world datasets, spanning a wide range of domains, sizes, and dimensionalities.[1] We implemented Reap in C++, ran experiments on a 12-Core Intel Xeon E5-2643 CPU, and report wall clock time.

5.5.1 Verifying the Relaxation on Synthetic Data

First, we test and verify our relaxation on data with known ground truth. To do so, we generate synthetic data. In each trial, we generate a random dataset X of $4\,096$ rows over 256 attributes. Firstly, we randomly generate and insert $2\,048$ characteristic patterns into S^*. Then, into each row, we randomly insert patterns from S^* using their corresponding frequency. Lastly, we introduce additive noise by randomly inserting items into each row, independently and with a probability of 5%. In total, we sample 20 synthetic datasets and compare $\tilde{p}$ with p^*. Even though we have access to the true pattern sets S^*, the computation of the true likelihood is intractable for p^*. Therefore, we compare the divergence between p^* or $\tilde{p}$ and empirical frequencies q for S^*.

In Figure 5.3, we see that p^* cannot insert more than 40–80 patterns from S^* for $\beta = 12$ without considerable runtime cost, on average over all trials. For the same budget, we observe the exponential inference time growth for the relaxation $\tilde{p}$ significantly later, at around $1\,800$ patterns. On real-world data, we usually observe a discovered pattern set with a size in the order of tens to hundreds, but we show that the scalability of $\tilde{p}$ is sufficient to handle even significantly bigger cases for budgets deemed large in practice.

On the left, we see that for smaller β, the true distribution p^* converges early, because there are no factors left that are below the budget. However, in this experiment, $\tilde{p}$ can handle the ground-truth patterns S^* for a budget of up to 12 patterns per factor, and it is capable of reaching the minimal divergence of 0. For $\tilde{p}$, we can further observe that strictly limiting the budget does not have a significant impact on

[1] https://archive.ics.uci.edu/ml, http://fimi.ua.ac.be/data/

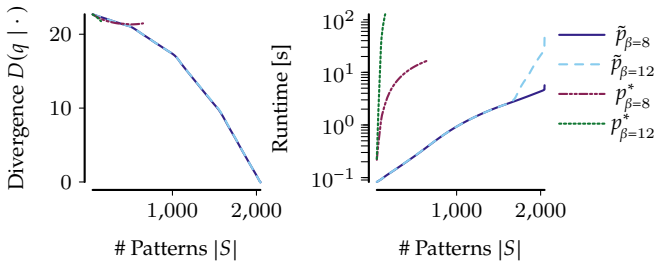


Figure 5.3: RelEnt approximates the maximum entropy distribution well and is much faster to infer on synthetic data. We show the divergence of $\tilde{p}$ and p^* to empirical frequencies q (left) and the elapsed time of inferring the likelihood for different β on synthetic data for increasing $|S|$ (right).

the divergence between q and the relaxation $\tilde{p}$. This is due to the fact that $\tilde{p}$ is overall less constrained than p^*, even for the same β.

5.5.2 Verifying the Relaxation on Real-World Data

Now, we test and verify $\tilde{p}$ on real-world datasets. To do so, we first discover a pattern set S using Reap for each dataset. Then, we use the same set S to compute the distributions $\tilde{p}$ and p^*. We compare both distributions in terms of the likelihood ratios $\Lambda = \ell/\ell^{(0)}$ with regard to that S and its initial independence model $S^{(0)}$. Next, for $\beta = 12$, we compare the time it takes to compute the objective ℓ of $\tilde{p}$ versus p^*.

All datasets that we use in our experiments are publicly available. We take *Chess*, *Connect*, *Mushroom*, *Pumsb*, *Kosarak*, *Retail*, *Accidents* from the Itemset Mining Dataset Repository.[2] We remove stop words, lemmatize and binarize the *AGnews* text corpus, and for the *AGnews (Titles)* dataset, we only consider news titles.[3] Similarly, we lemmatize and binarize the two versions of the *CORD 19* dataset by extracting the abstracts from the CORD 19 open research dataset.[4] The *DQ* dataset of lemmatized Deep-Learning and Quantum-Theory ArXiv abstracts can be found in our online material. To reduce the number of attributes of the *Instacart* dataset, we combine products from the same category, e.g.,

[2]fimi.ua.ac.be/data
[3]di.unipi.it/~gulli/AG_corpus_of_news_articles
[4]kaggle.com/allen-institute-for-ai/CORD-19-research-challenge

we merge Thin Spaghetti with Regular Spaghetti into the Spaghetti meta-category.[5] We use the *Chainstore* dataset from the SPMF dataset collection[6]. All remaining datasets are from the UCI Machine Learning Repository.[7] In Table 5.1, we provide basic statistics for the datasets and state the minimum support we use in our experiments.

Table 5.1: For each dataset, we show the number of rows ($|X|$), items ($\dim X$), groups, density, and the mean transaction length.

Dataset	$\lvert X\rvert$	$\dim X$	$\mathbb{E}_{x\in X}[\lvert x\rvert]$	density	groups
Higgs	11000000	247	28.00 ± 0.00	0.1133	2
SUSY	5000000	178	18.00 ± 0.00	0.1011	2
Instacart	2620570	1235	3.14 ± 2.18	0.0025	1
Chainstore	1112949	46086	7.23 ± 8.91	0.0002	1
POWER C	1040000	125	7.00 ± 0.00	0.0560	1
KDD Cup 99	1000000	135	16.00 ± 0.00	0.1185	1
PAMAP	1000000	82	23.93 ± 0.73	0.2919	1
Kosarak	990002	41270	8.10 ± 23.62	0.0002	1
Covtype	581012	64	11.95 ± 0.23	0.1866	2
Record Link	574913	27	10.00 ± 0.00	0.3704	1
Accidents	340183	468	33.81 ± 2.94	0.0722	1
COD RNA	271617	16	8.00 ± 0.00	0.5000	2
Skin	245057	12	4.00 ± 0.06	0.3330	1
AG Headlines	127600	5243	3.09 ± 1.49	0.0006	4
AG News	127600	11489	13.63 ± 4.05	0.0012	4
Retail	88162	16470	10.31 ± 8.16	0.0006	1
Connect	67557	129	42.00 ± 0.00	0.3256	3
BMS WV 1	59602	497	2.51 ± 4.85	0.0051	1
BMS WV 2	77512	3340	4.62 ± 6.07	0.0014	1
Pumsb	49046	2113	74.00 ± 0.00	0.0350	1
Adult	48842	97	13.87 ± 0.48	0.1430	2
Plants	34781	69	8.69 ± 13.11	0.1259	1
CORD 19	32915	3517	62.67 ± 31.77	0.0179	1
Chess	28056	51	6.00 ± 0.00	0.1176	18
Letter Recognition	20000	102	16.00 ± 0.00	0.1569	26
US Census	13369	392	68.00 ± 0.37	0.1735	1
Nursery	12960	30	8.00 ± 0.00	0.2667	5
Pen Digits	10992	76	16.00 ± 0.00	0.2105	10

[5]The Instacart Online Grocery Shopping Dataset 2017, accessed from instacart.com/datasets/grocery-shopping-2017
[6]philippe-fournier-viger.com/spmf
[7]archive.ics.uci.edu/ml

DQ	9993	434	22.30 ± 10.40	0.0514	1
Mushroom	8124	117	22.00 ± 0.00	0.1880	2
Breast Cancer	7325	397	11.67 ± 13.06	0.0294	2
Page Blocks	5473	39	10.00 ± 0.00	0.2564	5
DNA	5186	180	45.53 ± 5.22	0.2530	3
Waveform	5000	98	21.00 ± 0.00	0.2143	3
DNA Amplification	4587	391	5.78 ± 8.40	0.0148	1
Hypothyroid	3247	86	43.19 ± 0.39	0.5022	1
Led 7	3200	19	7.00 ± 0.00	0.3684	10
kr-vs-kp	3196	73	36.48 ± 0.50	0.4998	1
Splice	3190	287	60.73 ± 0.44	0.2116	1
Mammals	2183	121	24.81 ± 8.25	0.2050	1
German Credit	1000	110	38.70 ± 0.46	0.3518	1
Tic Tac Toe	958	27	9.74 ± 0.44	0.3606	1
Anneal	898	71	13.31 ± 1.45	0.1874	5
ICDM	859	3933	47.67 ± 14.32	0.0121	1
Diabetis	768	38	8.00 ± 0.00	0.2105	2
Australian Credit	653	124	51.53 ± 0.50	0.4155	1
Soybean	630	50	16.93 ± 0.25	0.3387	1
Vote	435	48	16.33 ± 0.47	0.3403	1
Ionosphere	351	155	34.00 ± 0.00	0.2194	2
Primary Tumor	336	31	15.79 ± 0.41	0.5092	1
Heart	303	50	12.98 ± 0.14	0.2596	5
Heart (Cleveland)	296	95	45.52 ± 0.50	0.4792	1
Audiology	216	146	67.13 ± 0.34	0.4598	1
Wine	178	65	13.00 ± 0.00	0.2000	3
Hepatitis	155	52	18.92 ± 1.83	0.3639	1
Iris	150	19	4.00 ± 0.00	0.2105	3
Lymph	148	68	27.72 ± 0.45	0.4077	1
Zoo	101	36	16.06 ± 0.24	0.4461	1

We show in Figure 5.4 that the likelihoods of S inferred by $\tilde{p}$ and p^* are, generally speaking, close to each other. If we use the relaxed $\tilde{p}$ instead of p^*, we observe a loss of at most 6% in likelihood ratio but also a gain of up to 9%. On approximately half of the datasets, our less constrained relaxation results in models with higher likelihood. In all but two cases, the inference by means of $\tilde{p}$ takes significantly less time than using p^*. However, these outlier datasets, *Iris* and *Breast Cancer*, are tiny, such that the absolute time is in the order of milliseconds and therefore negligible. In the majority of cases, our relaxation uses significantly less than 50% of the time taken by p^*.

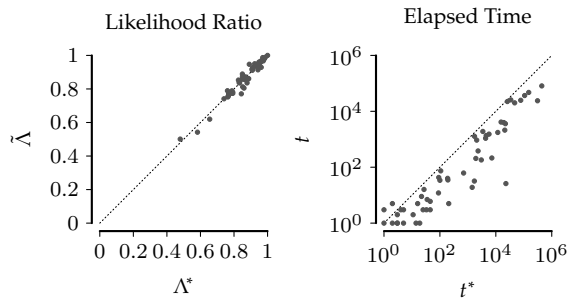


Figure 5.4: RelEnt approximates the maximum entropy distribution well and is faster to infer on real-world data. We compare $\tilde{p}$ and p^* using the likelihood ratio $\Lambda = \ell/\ell^{(0)}$ (left) (lower is better) and runtime (right) (milliseconds, log scale, lower is better) of inferring the likelihood using $\tilde{p}$ versus p^* for the same sets S, discovered by Reap on 57 real-world datasets using $\beta = 12$.

5.5.3 Reap as a Predictive Model

Generally speaking, it is very hard to compare different pattern sets. One way to objectively compare the quality is to evaluate how well these characterize and differentiate groups of data points. The better the statistic S models these groups, the better we can classify points, unless the groups have the same distribution. For a given statistic S_i and a set of factors $\mathcal{S}_i$ per group i, the predictive model of $\tilde{p}$ labels data points x

$$\arg\max_i \tilde{p}_i(x \mid \mathcal{S}_i)$$

with the label of the group under which x exhibits the largest likelihood, analogously for p^*.

In this experiment, we compare $\tilde{p}$ to p^* on 22 labeled datasets. We perform 10-fold cross-validation. In each fold, we randomly sample training (80%) and test data (20%), such that we preserve the relative sizes per group. Independently for each group in the training data, we estimate a model using Reap and Desc, respectively.

In Figure 5.5, we report the mean true positive rate (tpr) and average classification time for the sampled test datasets. We observe that in all but 2 cases, Reap has the same or better classification accuracy than Desc, while Reap always uses significantly less time to classify

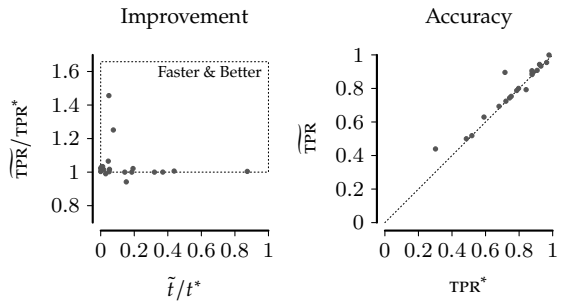


FIGURE 5.5: RELENT classifies better than the constrained maximum entropy distribution. We show the relative improvement (left) in accuracy $\widetilde{\text{TPR}}/\text{TPR}^*$ (higher is better) and relative runtime $\tilde{t}/t^*$ (lower is better), and we show the absolute true positive rate (higher is better) of $\tilde{p}$ versus p^* (right), for all 22 labeled datasets.

the datasets. Note that $\tilde{p}$ and p^* are generative models, and we trained neither to be specialized classifiers.

5.5.4 REAP FOR DISCOVERING THE COMPOSITION

Here, we investigate our relaxation in terms of decomposing a dataset and describing these components in terms of characteristic and common patterns, as first proposed by [43] (cf. Chapter 2). In a nutshell, DISC iteratively splits the dataset into components, assigns data points to the likeliest component, and characterizes these components using DESC. We replace DISC's distribution and pattern miner by $\tilde{p}$ and REAP, and call the result DĨSC.

In Fig 5.6, we show the likelihood ratio and runtime of DĨSC and DISC. We can see that DĨSC discovers compositions that have a likelihood similar to the result of DISC. However, DĨSC usually takes significantly less time than DISC. This is hardly surprising, as this tasks relies heavily on the inference of the distribution. Furthermore, in most cases in which DĨSC is slower, the composition has a higher likelihood.

5.5.5 REAP FOR MINING SETS OF PATTERNS

In this experiment, we evaluate the quality of REAP data summarization compared to other pattern mining methods. To do so, we discover

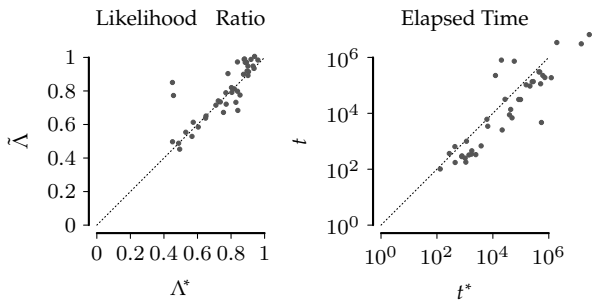


Figure 5.6: Reap discovers the composition efficiently and results in a high likelihood. We compare the likelihood ratio $\Lambda = \ell/\ell^{(0)}$ (left) (lower is better) and runtime (right) (milliseconds, log scale, lower is better) of Dīsc and Disc for 30 real-world datasets.

patterns using iim [60], opus [187, 189], mtv [118], Desc [43] and Reap. As mtv and Desc estimate the statically-factorized maximum entropy, we can fairly compare against them. On the other hand, iim and opus optimize for a different score. To include their results, we create a maximum entropy distribution of their patterns sets and use that to infer the likelihood. Creating a static factorization from their models would, however, exclude many of their patterns from the statistics. Thus, for a fair comparison, we use our relaxation and estimate $\tilde{p}$ from their patterns. opus discovers the top-k self-sufficient itemsets for a user-defined k, that is, we set k to the number of patterns that Reap has discovered. For iim, we limit the number of iterations and EM steps as done by the authors [60].

In Figure 5.7a, we show that Reap outperforms opus, iim, and mtv, and it is almost always within $\pm 5\%$ of the likelihood of Desc. In many experiments, we observe that the modeling power of Reap results in more patterns and a higher likelihood in comparison to the statically factorized mtv or Desc. In Figure 5.7b, we illustrate that the pattern sets discovered by mtv, Desc, iim, and Reap are similarly concise.

Now we take a closer look at the pattern sets and quantify how similar the discovered patterns are to one another. To do so, we measure the *average intra-pattern-set symmetric difference*,

$$\sigma(S) = \left|\binom{S}{2}\right|^{-1} \sum_{xy \in \binom{S}{2}} |x \triangle y| \ .$$

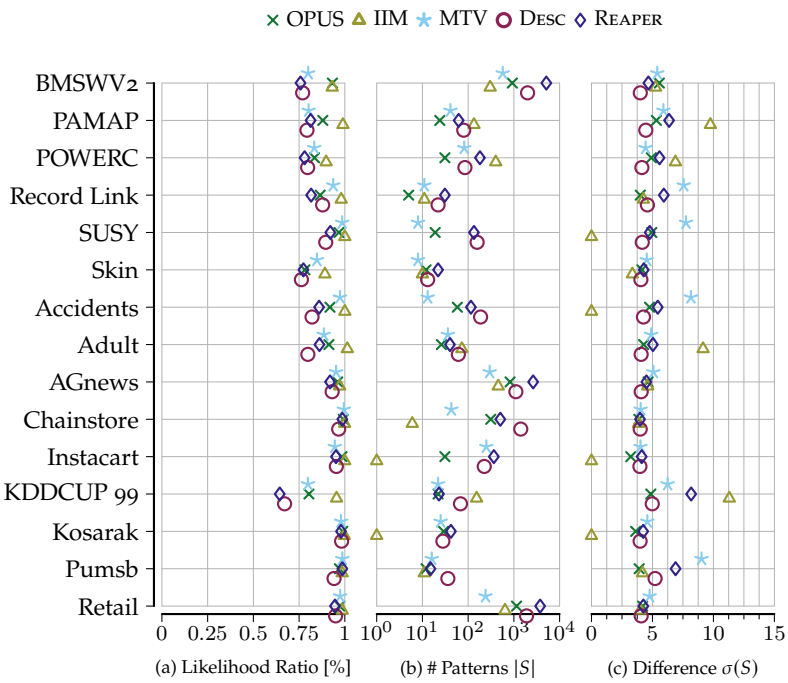


Figure 5.7: Reap efficiently discovers concise non-redundant pattern sets. We show the likelihood ratio (lower is better) in (a), the number of discovered patterns in (b), and the intra-pattern-set symmetric difference σ of pattern sets discovered in the 15 largest datasets in (c).

The higher $\sigma(S)$, the more different the patterns look on average. In Figure 5.7c, we see that Reap discovers patterns that are, on average, slightly more diverse than the results of Desc, but in general, the diversity is on par with the state of the art.

5.5.6 Qualitative Study

To conclude our experiments, we study the interpretability of the results produced by Reap via qualitative evaluation. To this end, we manually inspect the patterns Reap discovers in 3 datasets. The *DQ* dataset consists of 10 000 abstracts crawled from arXiv [43]. Half of the abstracts are from papers on Deep Learning, the other half are from papers on Quantum Physics. The *CORD-19* dataset was generated by extracting 33 000 abstracts from the original CORD-19 paper collec-

tion [185]. The *AGnews*[8] dataset consists of 127 600 news articles from 4 different categories. From all corpora, we remove stop words, extract and lemmatize nouns, verbs, and adjectives, and erase words with a frequency below 0.01.

In Table 5.2, we give a number of exemplary patterns discovered by Reap. The pattern set includes, for example, *magnetic field* and *computer vision* for DQ, *potential impact respiratory mechanism* and *antigen necessary* for CORD-19, and *International Space Station* for AGnews.

5.6 Conclusion

We introduced the relaxed maximum entropy distribution based on a generalized, dynamic factorization. This factorization trades inference complexity with information loss and results in a distribution that has higher statistical modeling power than previously-used exact models. On top of that, we provided an efficient and practical instantiation of this factorization based on set-cover principles. We formally linked the problem of estimating the relaxed distribution to the problem of discovering associations from data, for which we proposed the Reap algorithm that jointly discovers patterns and creates factors efficiently.

Experimentally, we have shown, on synthetic and real-world data, that the inference of the relaxed distribution is efficient and scalable, and that the relaxed distribution approximates the reference distribution well, without being similarly constrained. We extensively studied Reap in the context of multiclass classification and the discovery of the data composition, in which we performed at least equally well as competitors, however faster. Lastly, we compared Reap to pattern set miners, and we showed that our results are easily interpretable. This leads us to a comfortable conclusion: To find insightful patterns, it helps to relax.

[8] www.di.unipi.it/~gulli/AG_corpus_of_news_articles.html

Table 5.2: Reap discovers interpretable pattern sets. We show a selection of patterns discovered by Reap in 3 different datasets.

Dataset	Patterns (Selection)
CORD-19	*coronavirus acute severe respiratory syndrome sars* *patient hospital intensive care unit* *basic case reproduction number* *protective personal equipment* *PEDv epidemic diarrhea swine* *world health organization* *development target drug* *cell membrane protein fusion* *international preprint license copyright holder*
DQ	*deep convolutional neural network* *neural network adversarial attack* *stochastic gradient descent* *variational model inference* *quantum matrix density* *quantum space hilbert* *measurement quantum mechanic* *paper experimental result* *image classification task*
AGnews	*initial public offering,* *International Space Station,* *international agency nuclear atomic energy* *international space station astronaut* *world fastest supercomputer* *european union trade organization* *european space agency* *european commission brussels* *prime minister party coalition* *profit higher quarterly* *chief executive company chairman* *wall street stock* *gold medal olympics*

6
Efficiently Factorizing Boolean Matrices using Proximal Gradient Descent

In the previous chapter, we saw how relaxing the maximum entropy distribution can speed up the discovery of pattern sets in binary tabular data. But as pattern-set miners provide a high level of detail, exploring an exponentially-sized search space at a considerable computational cost, they struggle to report meaningful results on very large or very high-dimensional datasets that easily exceed multiple hundred thousand columns. The quest for scalable methods that can discover groups in tabular data and express them in terms of insightful patterns quickly leads to matrix factorization methods like Non-Negative Matrix Factorization (NMF) and Principal Component Analysis (PCA). These methods are not only scalable, but they also yield highly interpretable results—unless the data is Boolean, which is precisely our case of interest. Addressing the interpretability problem of NMF on Boolean data, Boolean Matrix Factorization (BMF) uses Boolean algebra to decompose the input into low-rank Boolean factor matrices. These matrices are highly interpretable and very useful in practice, but they come at the high computational cost of solving an NP-hard combinatorial optimization problem. Again, our strategy to reduce the computational burden is to relax. That is, we decompose Boolean matrices using a Boolean product of Boolean matrices obtained via a relaxed optimization scheme using linear algebra of intermediate

This chapter is based on the publication: Dalleiger and Vreeken [41].

continuous matrices. We therefore call this problem Boolean matrix factorization.

We propose to continuously relax BMF using a novel elastic-binary regularizer, from which we derive a proximal gradient algorithm, thus allowing us to use concepts from linear algebra instead of Boolean algebra. Through an extensive set of experiments, we demonstrate that our method works well in practice: On synthetic data, we show that it converges quickly, recovers the ground truth precisely, and estimates the simulated rank exactly. On real-world data, we improve upon the state of the art in recall, loss, and runtime, and a case study from the medical domain confirms that our results are easily interpretable and semantically meaningful.

6.1 Introduction

Discovering groups in data and expressing them in terms of common concepts is a central problem in many scientific domains and business applications, including cancer genomics [103], neuroscience [69], and recommender systems [80]. This problem is often addressed using variants of *matrix factorization*, a family of methods that decompose the target matrix into a set of typically low-rank factor matrices whose product approximates the input well. Prominent examples of matrix factorization are Singular Value Decomposition (SVD) [64], Principal Component Analysis (PCA) [64], and Nonnegative Matrix Factorization (NMF) [99, 100, 136]. These methods differ in how they constrain the matrices involved: SVD and PCA require orthogonal factors, while NMF constrains the target matrix and the factors to be nonnegative.

SVD, PCA, and NMF achieve interpretable results—unless the data is Boolean, which is ubiquitous in the real world. In this case, their results are hard to interpret directly because the input domain differs from the output domain, such that post-processing is required to extract useful information. *Boolean Matrix Factorization* (BMF) addresses this problem by seeking two low-rank *Boolean* factor matrices whose Boolean product is close to the Boolean target matrix [127]. The output matrices, now lying in the same domain as the input, are interpretable

and useful, but they come at the computational cost of solving an NP-hard combinatorial optimization problem [125, 127, 135]. To make BMF applicable in practice, we need efficient approximation algorithms.

There are many ways to approximate BMF—for example, by exploiting its underlying combinatorial or spatial structure [19, 20, 127], using probabilistic inference [153–155], or solving the related bi-clustering problem [131, 132]. Although these approaches achieve impressive results, they fall short when the input data is large and noisy. Hence, we take a different approach to overcome BMF's computational barrier. Starting from an NMF-like optimization problem, we derive a continuous relaxation of the original BMF formulation that allows intermediate solutions to be real-valued. Inspired by the elastic-net regularizer [201], we introduce the novel *elastic binary (Elb) regularizer* to regularize toward Boolean factor matrices. We obtain an efficient-to-compute *proximal operator* from our Elb regularizer that projects relaxed real-valued factors towards being Boolean, which allows us to leverage fast gradient-based optimization procedures. In stark contrast to the state of the art [74–76], which requires heavy post-hoc post-processing to actually achieve Boolean factors, we ensure a Boolean outcome upon convergence by gradually increasing the projection strength using a *regularization rate*. We combine our relaxation, efficient proximal operator, and regularization rate into an *Elastic Boolean Matrix Factorization* algorithm (Elbmf) that scales to large data, results in accurate reconstructions, and does so without relying on heavy post-processing procedures. Elb and its rate are, however, not confined to BMF and can regularize, e.g., binary MF or bi-clustering [76].

In summary, our main contributions are as follows:

1. We introduce the Elb regularizer.
2. We overcome the computational hardness of BMF leveraging a novel relaxed BMF problem.
3. We efficiently solve the relaxed BMF problem using an optimization algorithm based on proximal gradient descent.

The remainder of this chapter proceeds as follows. In Section 6.2, we formally introduce the BMF problem and its relaxation, define our Elb regularizer and its proximal point operator, and show how to

ensure a Boolean outcome upon convergence. We discuss related work in Section 6.3, validate our method through an extensive set of experiments in Section 6.4, and conclude with a discussion in Section 6.5.

6.2 Theory

Our goal is to factorize a given Boolean target matrix into at least two smaller, low-rank Boolean factor matrices, whose product comes close to the target matrix. Since the factor matrices are Boolean, this product follows the algebra of a Boolean semi-ring, i.e., it is identical to the standard outer product on a field where addition obeys $1 + 1 = 1$. We define the product between two Boolean matrices $\mathrm{U} \in \{0,1\}^{n \times k}$ and $\mathrm{V} \in \{0,1\}^{k \times m}$ on a Boolean semi-ring $(\{0,1\}, \vee, \wedge)$ as

$$[\mathrm{U} \circ \mathrm{V}]_{ij} = \bigvee_{l \in [k]} \mathrm{U}_{il} \mathrm{V}_{lj} \ ,$$

where $\mathrm{U} \in \{0,1\}^{n \times k}$, $\mathrm{V} \in \{0,1\}^{k \times m}$, and $\mathrm{U} \circ \mathrm{V} \in \{0,1\}^{n \times m}$. This gives rise to the BMF problem.

Problem 6.1 (Boolean Matrix Factorization). For a given target matrix $\mathrm{A} \in \{0,1\}^{n \times m}$, a given matrix rank $\mathbb{N} \ni k \leq \min\{n, m\}$, and $A \oplus B$ denoting logical exclusive or, discover the factor matrices $\mathrm{U} \in \{0,1\}^{n \times k}$ and $\mathrm{V} \in \{0,1\}^{k \times m}$ that minimize

$$\|\mathrm{A} - \mathrm{U} \circ \mathrm{V}\|_F^2 = \sum_{ij} \mathrm{A}_{ij} \oplus [\mathrm{U} \circ \mathrm{V}]_{ij} \ . \tag{6.2.1}$$

While beautiful in theory, this problem is NP-complete [125]. Thus, we cannot solve it exactly for all but the smallest matrices. In practice, we hence have to rely on approximations. Here, we relax the Boolean constraints of Eq. (6.2.1) to allow non-negative, *non-Boolean* 'intermediate' factor matrices during the optimization, allowing us to use linear algebra, rather than Boolean algebra. In other words, we solve the non-negative matrix factorization (NMF) problem [136]

$$\|\mathrm{A} - \mathrm{UV}\|_F^2 \ ,$$

subject to $U \in \mathbb{R}_+^{n \times k}$ and $V \in \mathbb{R}_+^{k \times m}$. In contrast to the original BMF formulation, we can approximate this problem efficiently, e.g., via a Gauss-Seidel scheme. Although efficient, using plain NMF, however, disregards the Boolean structure of our matrices and produces factor matrices from a different domain, which are consequently hard to interpret and potentially very dense. To benefit from efficient optimization and still arrive at Boolean outputs, we allow real-valued intermediate solutions and regularize them towards becoming Boolean.

To steer our optimization towards Boolean solutions, we penalize non-Boolean solutions using a regularizer. This idea has been explored in prior work. There exists the l_1-inspired Primp regularizer [74], which is

$$-\kappa[-|1-2x|+1]$$

for values inside $[0, 1]$ and ∞ otherwise, and the l_2-inspired bowl-shaped regularizer [199], which is

$$\lambda(x^2 - x)^2/2$$

everywhere on the real line. Although both have been successfully applied to BMF, both also have undesirable properties: The Primp regularizer penalizes well *inside* the interval $[0, 1]$ but is non-differentiable on the outside, while the bowl-shaped regularizer is differentiable and penalizes well *outside* the interval $[0, 1]$ but is almost flat on the inside. Hence, both regularizers are problematic if used individually. Combining them, however, yields a regularizer that penalizes non-Boolean values well across the full real line. To combine l_1-regularization and l_2-regularization, we use the *elastic-net regularizer*,

$$r(x) = \kappa\|x\|_1 + \lambda\|x\|_2^2 \,,$$

which, however, only penalizes *non-zero* solutions [201]. To penalize *non-Boolean* solutions, we combine two elastic-net regularizers into our (almost W-shaped) *Elb regularizer*,

$$R(X) = \sum_{x \in X} \min\{r(x), r(x-1)\} \,, \tag{6.2.2}$$

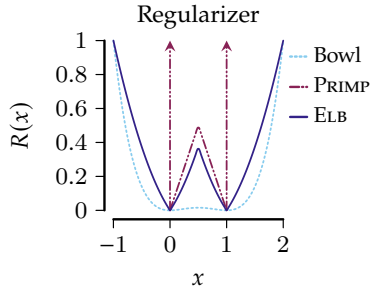


Figure 6.1: Elb regularizes non-Boolean values well. We show the three regularizers Bowl, Primp, and Elb, for $\lambda = \kappa = 0.5$, and see that only our Elb regularizer penalizes non-Boolean values well.

where $X \in \{\mathrm{U}, \mathrm{V}\}$. In Figure 6.1, we show all three regularizers in the range of $[-1, 2]$, for $\lambda = \kappa = 0.5$. We see that only the Elb regularizer penalizes non-Boolean solutions across the full spectrum, and summarize our regularized relaxed BMF as follows.

Problem 6.2 (Elastic Boolean Matrix Factorization). For a given target matrix $\mathrm{A} \in \{0,1\}^{n \times m}$ and a given matrix rank $\mathbb{N} \ni k \leq \min\{n, m\}$, discover the factor matrices $\mathrm{U} \in \mathbb{R}_+^{n \times k}$ and $\mathrm{V} \in \mathbb{R}_+^{k \times m}$ that minimize

$$\|\mathrm{A} - \mathrm{UV}\|_F^2 + R(\mathrm{U}) + R(\mathrm{V}) \ .$$

Although this is a relaxed problem, it is still non-convex, and therefore, we cannot solve it straightforwardly. The problem, however, is suitable for the Gauss-Seidel optimization scheme. That is, we alternatingly fix one factor matrix to optimize the other. By doing so, we generate a sequence

$$\begin{aligned} \mathrm{U}_{t+1} &\leftarrow \arg\min_{\mathrm{U}} \|\mathrm{A} - \mathrm{UV}_t\|_F^2 + R(\mathrm{U}) \ , \\ \mathrm{V}_{t+1} &\leftarrow \arg\min_{\mathrm{V}} \|\mathrm{A} - \mathrm{U}_{t+1}\mathrm{V}\|_F^2 + R(\mathrm{V}) \ , \end{aligned} \tag{6.2.3}$$

of simpler-to-solve sub-problems, until convergence. Now, each sub-problem is again a sum of two $f(X)+R(X)$ functions, where f is the loss $\|\cdot\|_F^2$, and $R(X)$ is the regularizer. This allows us to follow a proximal

gradient approach, i.e., we use *Proximal Alternating Linear Minimization* (PALM) [24, 144]. In a nutshell, we minimize a sub-problem by following the gradient ∇f of f, to then use the proximal operator for R to nudge its outcome toward a Boolean solution. That is, for the gradients $\nabla_U f = UVV^\top - AV^\top$ and $\nabla_V f = U^\top UV - U^\top A$, we compute the step

$$\operatorname{prox}_R(X - \eta \nabla f) \,, \tag{6.2.4}$$

where η represents the step size, which we compute in terms of Lipschitz constant, rather than relying on a costly line search [24]. To further improve the convergence properties, we make use of an inertial term that linearly combines X_t with X_{t-1} before applying Eq. (6.2.4) (see [144] for a detailed description). We now derive the proximal operator for the Elb regularizer, before discussing how we ensure that the factor matrices are Boolean, and summarizing our approach as an algorithm.

6.2.1 Proximal Mapping

To solve the sub-problem

$$\arg\min_X \ell_{\kappa\lambda}(X) \text{ for } \ell_{\kappa\lambda}(X) = f(X) + R(X)$$

from Eq. (6.2.3) for $X \in \{U, V\}$, we need a proximal operator [139] that projects values towards a regularized point.

To do this, we derive a proximal operator from the Elb (Eq. (6.2.2), repeated below)

$$R(X) = \sum_{x \in X} \min\{r(x), r(x-1)\} \,.$$

Starting with the definition [139] of the general proximal operator

$$\arg\min_Y \frac{1}{2}\|X - Y\|_2^2 + R(Y) \,,$$

we observe that this proximal operator is coordinate-wise solvable. This allows us to derive a *scalar* proximal operator, which we then apply to each value in the matrix independently. Substituting the regularizer

R with its scalar version, we obtain a scalar proximal operator

$$\operatorname{prox}_{\kappa\lambda}(x) = \arg\min_{y\in\mathbb{R}} \frac{1}{2}(y-x)^2 + \min\{r(y), r(y-1)\} \ , \qquad (6.2.5)$$

which is non-convex, has no unique minima, and, therefore, is not straightforwardly solvable. We can, however, separate this function into two *locally convex* V-shaped functions, which are straightforwardly solvable. By asserting that its least-squares solution is either at most $1/2$ (if $x \leq 1/2$), or greater than $1/2$ (if $x > 1/2$), we can address each case independently, and merge the outcome into a single piecewise proximal operator.

Case I In the first case, we address the operator for $x \leq 1/2$. For this, we start by simplifying our scalar proximal operator Eq. (6.2.5), by substituting $r(y)$ with its definition, and get

$$\operatorname{prox}^{\leq 1/2}_{\kappa\lambda}(x) = \frac{1}{2}(y-x)^2 + \frac{\lambda'}{2}y^2 + \kappa|y| \ ,$$

for $\lambda'/2 = \lambda$. Then, we take its partial derivative for y

$$\frac{\partial}{\partial y} \operatorname{prox}^{\leq 1/2}_{\kappa\lambda}(x) = (y-x) + \lambda' y + \kappa \operatorname{sign}(y) \ ,$$

which we set to zero, obtaining

$$\begin{gathered} 0 = (y-x) + \lambda' y + \kappa \operatorname{sign}(y) \\ \Leftrightarrow \\ 0 = y(1+\lambda') - x + \kappa \operatorname{sign}(y) \ . \end{gathered}$$

By asserting that we can obtain a better least-squares solution if y has the same sign as x, we can substitute the sign of x with $\operatorname{sign}(y)$, and get

$$\begin{gathered} 0 = y(1+\lambda') - x + \kappa \operatorname{sign}(x) \\ \Leftrightarrow \\ y = (1+\lambda')^{-1}[x - \kappa \operatorname{sign}(x)] \ , \end{gathered}$$

which concludes the first case.

Case II Analogously, we now repeat the steps from above for $x > 1/2$. Again, we start by simplifying Eq. (6.2.5), substituting $r(y-1)$

$$\operatorname{prox}_{\kappa\lambda}^{>1/2}(x) = \frac{1}{2}(y-x)^2 + \frac{\lambda'}{2}(y-1)^2 + \kappa|y-1| \ ,$$

for $\lambda'/2 = \lambda$. By taking its partial derivative for y

$$\frac{\partial}{\partial y} \operatorname{prox}_{\kappa\lambda}^{>1/2}(x) = (y-x) + \lambda'(y-1) + \kappa \operatorname{sign}(y-1) \ ,$$

and setting it to zero, we obtain

$$\begin{aligned} 0 &= (y-x) + \lambda'(y-1) + \kappa \operatorname{sign}(y-1) \\ &\Leftrightarrow \\ 0 &= (1+\lambda')y - \lambda' - x + \kappa \operatorname{sign}(y-1) \ . \end{aligned}$$

Then, asserting that the least-squares solution does not get worse by using the same sign for $y-1$ and $x-1$, we can substitute the sign of $x-1$ with $\operatorname{sign}(y-1)$, and get

$$\begin{aligned} 0 &= y(1+\lambda') - x - \lambda' + \kappa \operatorname{sign}(x-1) \\ &\Leftrightarrow \\ y &= (1+\lambda')^{-1}[x - \kappa \operatorname{sign}(x-1) + \lambda'] \ , \end{aligned}$$

which concludes the $x > 1/2$ case.

Combining Case I & Case II Combining the cases above yields our piecewise proximal operator

$$\operatorname{prox}_{\kappa\lambda}(x) \equiv (1+\lambda)^{-1} \begin{cases} x - \kappa \operatorname{sign}(x) & \text{if } x \leq \frac{1}{2} \\ x - \kappa \operatorname{sign}(x-1) + \lambda & \text{otherwise} \ . \end{cases} \tag{6.2.6}$$

Alternative Proximal Operator Considering Eq. (6.2.5), we notice that the term $y-x$ is squared, which means that there are multiple solutions to this equation. We derive the alternative operator analogously to the

steps above, however, by switching the positions of y and x in f.

$$\operatorname{prox}^{\text{alt.}}_{\kappa\lambda}(x) \equiv (\lambda - 1)^{-1} \begin{cases} -x - \kappa \operatorname{sign}(x) & \text{if } x \leq \frac{1}{2} \\ -x - \kappa \operatorname{sign}(x-1) + \lambda & \text{otherwise .} \end{cases} \tag{6.2.7}$$

Since this operator is denominated by $\lambda' - 1$, we need to ensure that $\lambda' \neq 1$. Because our original proximal operator in Eq. (6.2.6) is denominated by $\lambda' + 1$, and since λ' is usually positive, we are not required to take extra precautions. Since this is more convenient, we select Eq. (6.2.6) as our proximal operator, rather than taking extra precautions when using Eq. (6.2.7).

Algorithm 6.1: **Elbmf**

Input: Matrix $\mathrm{A} \in \{0,1\}^{n \times m}$, rank $k \in \mathbb{N}$
Output: Factors $\mathrm{U} \in \{0,1\}^{n \times k}$, $\mathrm{V} \in \{0,1\}^{k \times m}$

```
initialize U, V uniformly at random
for t = 1, 2, ... until convergence
    U ← arg reduce_U ℓ_{κλ_t}(A, U, V)
    V ← arg reduce_V ℓ_{κλ_t}(A, U, V)
```

6.2.2 Ensuring Boolean Factors

Our proximal operator only nudges the factor matrices toward *becoming* Boolean. We, however, want to ensure that our results *are* Boolean. To this end, the state-of-the-art method Primp relies heavily on post-processing, performing a very expensive joint two-dimensional grid search to guess the 'best' pair of rounding thresholds, which are then used to produce Boolean matrices. Although this tends to work in practice, it is an inefficient post-hoc procedure—and thus, it would be highly desirable to have Boolean factors already upon convergence. To achieve this without rounding or clamping, we revisit our regularizer, which binarizes more strongly if we regularize more aggressively. Consequently, if we regularize too aggressively, we converge to a sub-optimal solution, and if we regularize too mildly, we do not binarize our solutions. To prevent subpar solutions and still binarize our out-

put, we start with a weak regularization and gradually increase its strength.

Considering Eq. (6.2.6), we see that a stronger regularization increases the distance over which our proximal operator projects. Thus, if we set the l_1-distance controlling κ too high, we will immediately leap to a Boolean factor matrix, which will terminate the algorithm and yield a suboptimal solution. Regulating the l_2-distance controlling λ is a less delicate matter. Hence, we gradually increase λ to prevent a subpar solution and achieve a Boolean outcome, using a *regularization rate*

$$\lambda_t = \lambda \cdot \nu_t \quad \text{for} \quad \nu_t \geq 0 \quad \forall t \geq 0$$

that gradually increases the proximal distance at a user-defined rate. If Elbmf stops without convergence, we bridge the remaining integrality gap by projecting the outcome onto its closest Boolean counterpart, using our proximal operator (see Figure 6.2).

We summarize the considerations laid out above as the generic version of Elbmf in Algorithm 6.1 or as the iPALM-based version in Algorithm 6.2. The computational complexity of Elbmf is bounded by the complexity of computing the gradient, which is identical to the complexity of matrix multiplication. Therefore, for all practical purposes, Elbmf is sub-cubic $O\left(n^{2.807}\right)$ using Strassen's algorithm.

6.3 Related Work

Matrix factorization is a well-established family of methods, whose members, such as SVD, PCA, or NMF, are used everywhere in machine learning. Almost all matrix factorization methods operate on real-valued matrices, however, while BMF operates under Boolean algebra. *Boolean Matrix Factorization* originated in the combinatorics community [129] and was later introduced to the data mining community [127], where many cover-based BMF algorithms were developed [19, 20, 126, 127].

In recent years, BMF has gained traction in the machine learning community, which tends to tackle the problem differently. Here, *relaxation-based approaches* that optimize for a relaxed but regularized

Algorithm 6.2: **Elbmf** (using iPALM [144])

Input:

Target Matrix	$\mathrm{A} \in \{0,1\}^{n \times m}$
Rank	$k \in \mathbb{N}$,
l_1 Regularizer Coefficients	$\kappa \in \mathbb{R}$,
l_2 Regularizer Coefficients	$\lambda \in \mathbb{R}$,
Regularization Rate	$\nu_t \in \mathbb{N} \to \mathbb{R}$,
optional Inertial Parameter	$\beta \in \mathbb{R}_+$

Output: Factors $\mathrm{U} \in \{0,1\}^{n \times k}$, $\mathrm{V} \in \{0,1\}^{k \times m}$

$\mathrm{U}_0 = \mathrm{U}_1 \leftarrow \mathrm{rand}(n, k)$
$\mathrm{V}_0 = \mathrm{V}_1 \leftarrow \mathrm{rand}(k, m)$
for $t = 1, 2, \ldots$ **until convergence**
  $\lambda_t \leftarrow \lambda \cdot \nu_t$
  $\mathrm{V} \leftarrow \mathrm{V}_t$
  $\mathrm{U} \leftarrow \mathrm{U}_{t-1} + \beta(\mathrm{U}_{t-1} - \mathrm{U}_{t-2})$
  $\nabla_{\mathrm{U}} f = \mathrm{U}\mathrm{V}\mathrm{V}^\top - \mathrm{A}\mathrm{V}^\top$
  $L \leftarrow \|\mathrm{V}\mathrm{V}^\top\|_2$
  $\mathrm{U} \leftarrow \mathrm{prox}_{\kappa L^{-1}, \lambda_t L^{-1}}\left(\mathrm{U} - L^{-1}\nabla_{\mathrm{U}} f\right)$
  $\mathrm{U}_t \leftarrow \mathrm{U}$
  $\mathrm{V} \leftarrow \mathrm{V}_{t-1} + \beta(\mathrm{V}_{t-1} - \mathrm{V}_{t-2})$
  $\nabla_{\mathrm{V}} f = \mathrm{U}^\top\mathrm{U}\mathrm{V} - \mathrm{U}^\top\mathrm{A}$
  $L \leftarrow \|\mathrm{U}^\top\mathrm{U}\|_2$
  $\mathrm{V} \leftarrow \mathrm{prox}_{\kappa L^{-1}, \lambda_t L^{-1}}\left(\mathrm{V} - L^{-1}\nabla_{\mathrm{V}} f\right)$
  $\mathrm{V}_t \leftarrow \mathrm{V}$
if U or V not Boolean *(if the loop aborted early (cf. Figure 6.2))*
  let $\lambda' \in \mathbb{R}$ be huge
  $\mathrm{U} \leftarrow \lfloor \mathrm{prox}_{0.5, \lambda'}(\mathrm{U}) \rceil$
  $\mathrm{V} \leftarrow \lfloor \mathrm{prox}_{0.5, \lambda'}(\mathrm{V}) \rceil$
return U, V

BMF [73, 74, 199] are related to our method, but they differ especially in their regularization. Hess et al. [74] introduce a regularizer that is only partially differentiable, and they rely heavily on post-processing to force a Boolean solution, and Zhang et al. [199] regularize only weakly between 0 and 1. In contrast, our regularizer penalizes well across the full spectrum and yields a Boolean outcome upon convergence. Building on a thresholding-based BMF formulation, Araujo et al. [9] also consider relaxations to benefit from gradient-based optimization.

Other recent approaches build on *probabilistic inference*. For example, Rukat et al. [153–155] combine Bayesian Modeling and sampling into their logical factor machine. A similar direction is taken by Ravanbakhsh et al. [149], who use graphical models and message passing, and Liang et al. [103], who combine MAP-inference and sampling. A different, *geometry-based approach* lies in locating dense submatrices by ordering the data to exploit the consecutive-ones property [173, 183]. Since BMF is essentially solving a bipartite graph partitioning problem, it is also closely related to Bi-Clustering and Co-Clustering [76, 131]. Neumann and Miettinen [132] use this relationship to efficiently solve BMF by means of a streaming algorithm. Although there are many different approaches to BMF, its biggest challenge to date remains scalability [125].

6.4 Experiments

We implement Elbmf in the Julia language and run experiments on 16 cores of an AMD EPYC 7702 and a single NVIDIA A100 GPU, reporting wall-clock time. We compare Elbmf with six other methods: four dedicated BMF methods (Asso [127], Grecond [20], OrM [155], and Primp [74]), one streaming Bi-Clustering algorithm (Sofa [132]), one elastic-net-regularized NMF method leveraging proximal gradient descent (NMF [99, 100, 136]), and one interpretable Boolean autoencoder (Binaps [56]). The code for Asso, Grecond, Primp, Sofa, OrM, and Binaps was written by their respective authors and is publicly available,[1] and we implement nmf in the Julia language.

[1] cs.uef.fi/~pauli/basso

Since NMF outputs non-negative factor matrices, rather than Boolean matrices, we cannot compare against NMF directly, so we clamp and round its solutions to the nearest Boolean outcome. To fairly compare against Binaps, we task it with autoencoding the target matrix as a reconstruction, given the matrix ranks from our experiments as the number of latent dimensions. We perform three sets of experiments. First, we ascertain that Elbmf works reliably on synthetic data. Second, we verify that it generally performs well on real-world data. And third, we illustrate that its outputs are semantically meaningful through an exploratory analysis of a biomedical dataset.

6.4.1 Performance of Elbmf on Synthetic Data

In the following experiments, we ask four questions: (**1**) How does Elbmf converge?; (**2**) How well does Elbmf recover the information in the target matrix?; (**3**) How consistently does Elbmf reconstruct low-density or high-density target matrices?; and (**4**) Does Elbmf estimate the underlying Boolean matrix rank correctly? To answer these questions, we generate synthetic data with known ground truth as follows. Starting with an all-zeros matrix, we randomly create rectangular, non-overlapping, consecutive areas of ones called *tiles*, each spanning a randomly chosen number of consecutive rows and columns, thus inducing matrices with varying densities. We then add additive noise by setting each cell to 1, uniformly at random, with varying noise probabilities.

How does Elbmf converge? To study how our method converges to a Boolean solution, we quantify relevant properties of the sequence of intermediate solutions (cf. Eq. (6.2.3)). First, to understand how quickly and stably Elbmf converges to a Boolean solution, we quantify

github.com/martin-trnecka/matrix-factorization-algorithms
bitbucket.org/np84/paltiling
cs.uef.fi/~pauli/bmf/sofa
eda.mmci.uni-saarland.de/prj/binaps
github.com/TammoR/LogicalFactorisationMachines

the *Boolean gap*,

$$\sum_{X \in \{U_t, V_t\}} |X|^{-1} \sum_{x \in X} \min\{|x|, |x-1|\} \ .$$

Second, to understand when we can safely round intermediate almost-Boolean solutions without losing information, we calculate, for the reconstruction B from *rounded* intermediate factors, the cumulative *Hamming process* as the fraction of bits that flip from iteration t to iteration $t+1$,

$$|A|^{-1} \|\mathrm{B}_t - \mathrm{B}_{t+1}\|_1 \ ,$$

and the *loss gap* as the difference between the relaxed loss and the loss from the *rounded* B.

As shown in Figure 6.2a, we achieve an almost-Boolean solution *without any rounding* after around 250 epochs, continuing until we reach a Boolean outcome. This is also the point at which the rounded intermediate solution and its relaxation are almost identical, as illustrated by the loss gap in Figure 6.2b. Considering the Hamming process in Figure 6.2c, we observe that Elbmf goes through an erratic bit-flipping phase in the beginning, followed by only minor changes in each iteration until iteration $t = 100$. Afterwards, Elbmf has settled on a solution—under our regularization rate. When using constant regularization instead, we continue to observe bit flips until the end of the experiment. Under constant regularization, the Boolean gap hardly decreases over time. Far from Boolean, the constant regularization thus also never closes the loss gap—which is unsurprising, given that its factors are less regularized. In other words, our regularization works well, and it allows us to safely binarize almost-converged factors that are ϵ-far from being Boolean by means of, e.g., our proximal operator.

How well does Elbmf recover the information in the target matrix? Having ensured that our method converges stably and quickly, we would like to assess whether it also converges to a high-quality factorization. To this end, we generate synthetic 40×30 matrices containing 5 random tiles each spanning 5 to 10 rows and columns, under additive noise levels between 0% (no noise) and 50%. We then compute the fraction

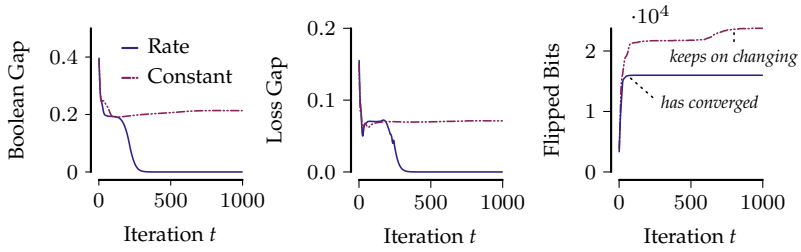


FIGURE 6.2: ELBMF converges quickly under a regularization rate. We report the progression over time of the *Boolean gap*, *loss gap*, and the *Hamming process*, for $1\,000$ iterations of ELBMF on synthetic 400×300 matrices with 10% noise and 5 random tiles covering between 50 and 100 rows or columns, under a constant regularization of $\lambda_t = 1$ or a regularization rate of $\lambda_t = 1.05^t$.

of *ones* in target A that is covered by the reconstruction $\mathrm{B} = \mathrm{U} \circ \mathrm{V}$, i.e., the *recall* (higher is better)

$$\|\mathrm{A}\|_1^{-1} \|\mathrm{A} \odot \mathrm{B}\|_1 \ .$$

To ensure that we fit the *signal* in the data, we additionally report the recall regarding the generating, noise-free ground-truth tiles A^*, denoted as recall*. Finally, to rate the overall reconstruction quality including *zeros*, we compute the *Hamming similarity* (higher is better) between the target matrix and its reconstruction

$$|\mathrm{A}|^{-1} \|\mathrm{A} - \mathrm{B}\|_1 \ .$$

We run each method on our synthetic datasets, targeting a matrix rank of 5. To account for random fluctuations, we average over 10 randomly drawn sets of 5 ground-truth tiles per 10% increment in noise probability. In Figure 6.3, we show similarity, recall, and recall*. We observe that in the noiseless case (0%), all methods except BINAPS recover the 5 ground-truth tiles with high accuracy, but only ASSO and ELBMF do so with perfect recalls. Starting with as little as 10% noise, both recalls of ASSO, GRECOND, SOFA, NMF, and BINAPS deteriorate quickly, while the similarity and both recalls of PRIMP and ELBMF remain high. In fact, ELBMF and PRIMP perform similarly across the board—which is highly encouraging, as unlike PRIMP, ELBMF does not

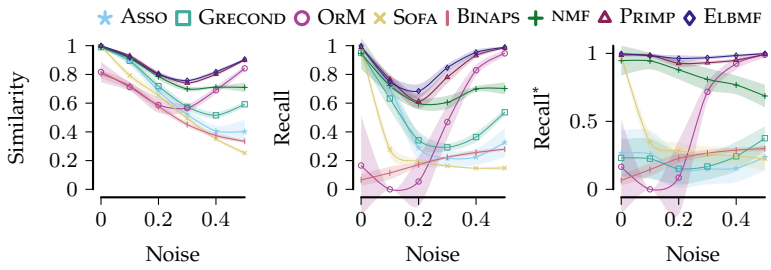


Figure 6.3: Overall, Elbmf reconstructs the noisy synthetic data well and recovers the ground-truth tiles. On synthetic data for additive noise levels increasing from 0% to 50%, we show mean as line and standard deviation as shade of *similarity*, *recall* w.r.t. the target matrix, and *recall** w.r.t. the noise-free ground-truth tiles, for Asso, Grecond, OrM, Sofa, Binaps, nmf, Primp, and Elbmf.

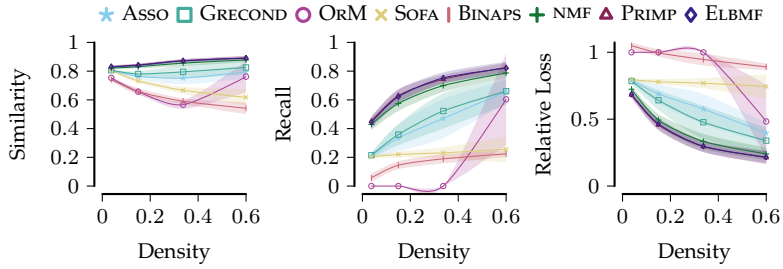


Figure 6.4: Elbmf reconstructs noisy synthetic high-density and low-density matrices consistently well. On synthetic data with fixed additive noise and increasing density, we show mean as line and standard deviation as shade of *similarity*, *recall*, and *relative loss*, for Binaps, Asso, Grecond, OrM, Sofa, Binaps, nmf, Primp, and Elbmf.

require post-processing. For Asso and Grecond, recall and similarity drop considerably, but they exhibit a slightly higher recall*. This means that these methods are robust against noise, but they fail to recover the remaining information. Starting low, OrM's recalls increase jointly with the noise level, suggesting that clean data is problematic for OrM. Reporting the standard deviations as the shaded region, we see little variance across all similarities—except for Asso and nmf in the highest-noise regime. The deviation of both recalls is, however, inconsistent for most methods, except for Binaps, Primp, and Elbmf. Overall, the performance characteristics of Elbmf are among the most reliable.

How robust is Elbmf regarding varying matrix densities?

To understand whether Elbmf performs consistently well on low-density and high-density matrices, we generate synthetic matrices as before, this time using fixed noise of 0.2, and varying the width and height of the ground-truth squared tiles from 3^2 to 12^2, resulting in densities between 0.0375 to 0.6, before noise.

In Figure 6.4, we show the similarity, recall, and loss of Binaps, Asso, Grecond, OrM, Sofa, nmf, Primp, and Elbmf. We can see that the increasing density affects the performance of all methods, however, it does not affect the performance of all methods *equally*. All methods—except OrM—improve in similarity, recall, and loss. With increasing density, OrM gets worse at first, before its loss shrinks significantly, such that it finishes outperforming Sofa and Binaps. From low to high density, Elbmf is the best-performing method across the board in similarity, recall, and loss.

So far, the synthetic data used in our experiments consisted only of *non-overlapping* tiles generated using rejection sampling. To obtain results on harder-to-separate data, we generate synthetic matrices as described previously—however, this time, allowing tiles to overlap arbitrarily by sampling without a rejection step. In Figure 6.5, we show similarity, recall, and recall* for the *overlapping case*, observing a behavior similar to Figure 6.3 across the board. Again, we notice the surprisingly good performance of rounded nmf reconstructions, outperforming Asso, Grecond, Sofa, and Binaps by large margins. Overall, Primp and Elbmf outperform Asso, Grecond, OrM, Sofa, Binaps, and nmf across varying noise levels in similarity, recall, and recall*.

As our last experiment on synthetic data, we ask whether our observations carry over to a low-noise scenario, in which the performance of Asso and Grecond improves significantly (cf. Figure 6.3). To answer this question, we study the effects of varying densities under a low noise level of only 5%, reporting the results in Figure 6.6. As tiny tiles are hard to distinguish from noise, we see an overall improvement with increasing density, regardless of the method. With less noise, Asso, Grecond, and nmf improve significantly in comparison to their

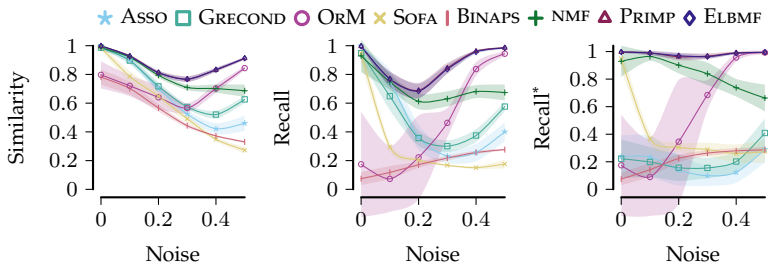


Figure 6.5: Elbmf reconstructs the noisy synthetic data well and recovers the ground-truth tiles also when the tiles are *overlapping*. On synthetic data for additive noise levels increasing from 0% to 50%, we show mean as line and standard deviation as shade of *similarity*, *recall* w.r.t. the target matrix, and *recall** w.r.t. the noise-free ground-truth tiles, for Binaps, Asso, Grecond, OrM, Sofa, nmf, Primp, and Elbmf.

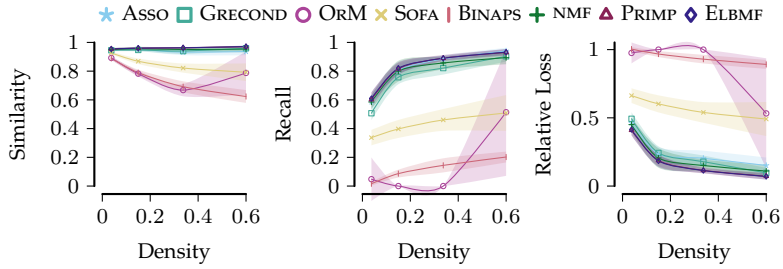


Figure 6.6: Elbmf reconstructs the low-noise synthetic high- and low-density matrices consistently well. On synthetic data with fixed additive noise level of as low as 5% and increasing density, we show mean as line and standard deviation as shade of *similarity*, *recall*, and *relative loss* w.r.t. the target matrix, for Binaps, Asso, Grecond, OrM, Sofa, nmf, Primp, and Elbmf.

performance under more noise (cf. Figure 6.4). They, however, are still outperformed by Primp and Elbmf in recall and loss. The similarities of Asso, Grecond, nmf, Primp, and Elbmf are close to 1, whereas Sofa, Binaps, and OrM exhibit lower similarity with increasing density. From Figure 6.4 and Figure 6.6, we see that Elbmf performs consistently well across varying densities, regardless of the noise level.

Overall, we observe that on synthetic data, Elbmf achieves best-in-class results for *overlapping* and *non-overlapping* tiles across all noise regimes: Elbmf, which does not use any post-processing, is consistently on par with its strongest competitor, which relies heavily on post-processing.

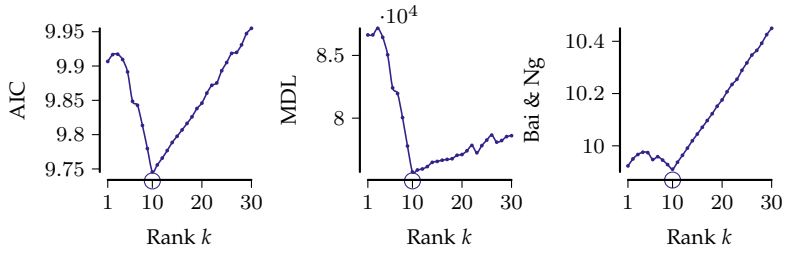


Figure 6.7: Using AIC, MDL, or Bai & Ng's first information criteria, Elbmf correctly detects the rank of the simulated 400×300 matrix of rank 10 to which we applied 10% additive noise.

Does Elbmf estimate the underlying Boolean matrix rank correctly? When a target rank is known, we can immediately apply Elbmf to factorize the data. In the real world, however, the target rank might be unknown. In this case, we need to *estimate* an appropriate choice from the data, and we use synthetic data to ensure that Elbmf does so correctly.

Since a higher matrix rank usually also means a better fit, selecting the best rank according to recall, loss, or similarity leads to overfitting—unless we properly account for the growth in model complexity. There are many *model selection criteria* that penalize complex models, such as AIC [7], Bai & Ng's criteria [8, 11], Nuclear-norm regularizing [67, 82], the information-theoretic Minimal Description Length principle (MDL) [66], or (Decomposed) Normalized Maximum Likelihood [81, 195]. Following common practice, and motivated by its practical performance in preliminary experiments, we choose MDL. That is, we select the *minimizer* of the sum of the log binomial $l(X) = \log \binom{|X|}{\|X\|_1}$ of the error matrix and the rows and columns of our factorization (assumed to be i.i.d.) [126],

$$l(\mathrm{A} \oplus [\mathrm{U} \circ \mathrm{V}]) + \textstyle\sum_{i \in [k]} l(\mathrm{U}_i^\top) + l(\mathrm{V}_i) + k \log(n \cdot m) \ .$$

To validate whether Elbmf recovers the correct rank, we synthetically generate a 400×300 matrix of ground-truth rank 10 with 10% additive noise. In Figure 6.7, we show AIC, MDL, and Bai & Ng's first criterion for each rank up to 30, finding that our method precisely discovers the right rank.

6.4.2 Performance of Elbmf on Real-World Data

Having ascertained that Elbmf works well on synthetic data, we turn to its performance in the real world. Here, we use 9 publicly available datasets[2] from different domains. To cover the *biomedical domain*, we extract the network containing empirical evidence of protein-protein interactions in *Homo sapiens* from the String database. From the Grand repository, we take the gene regulatory networks sampled from *Glioblastoma (GBM)* and *Lower Grade Glioma (LGG)* brain cancer tissues, as well as from non-cancerous *Cerebellum* tissue. The *TCGA* dataset contains binarized gene expressions from cancer patients, and we further obtain the single nucleotide polymorphism (SNP) mutation data from the $1k$ *Genomes* project, following processing steps from the authors of Binaps [56]. In the *entertainment domain*, we use the user-movie datasets *Movielens* and *Netflix*, binarizing the original 5-star-scale ratings by setting only reviews with more than 3.5 stars to 1. Finally, as data from the *innovation domain*, we derive a directed citation network between patent groups from patent citation and classification data provided by *PatentsView*. For each dataset with a given number of groups, such as cancer types or movie genres, we set the matrix rank k to 33 (TCGA), 28 (Genomes), 136 (Patents), 20 (Movielens), and 20 (Netflix). When the number of subgroups is unknown, we estimate the rank that minimizes MDL using Elbmf, resulting in 100 (GBM), 32 (LGG), 100 (String), and 450 (Cerebellum). We give basic statistics for all datasets in Table 6.1.

On *TCGA*, *Genomes*, *Movielens*, *Netflix*, and *Patents*, we set the L_2–regularizer $\lambda = 0.001$, the L_1-regularizer $\kappa = 0.005$, and the regularization rate to $\nu_t = 1.0033^t$. On *GBM*, *LGG*, and *Cerebellum*, we set the L_2-regularizer $\lambda = 0.001$, the l_1-regularizer $\kappa = 0.001$, and the regularization rate to $\nu_t = 1.0015^t$. We run nmf, Elbmf, Primp for at most $1\,500$ epochs on each dataset. In the case that Elbmf reaches its

[2] grand.networkmedicine.org
string-db.org
cancer.gov/tcga
internationalgenome.org
patentsview.org
grouplens.org/datasets/movielens
kaggle.com/datasets/netflix-inc/netflix-prize-data

Table 6.1: Our datasets are from different domains and cover a wide range of dimensionalities. We provide an overview of the real-world datasets involved in this study, listing their dimensionalities, densities, and selected target matrix ranks k (number of components) used in our experiments.

Dataset	Rank	Rows	Columns	Density
Genomes	28	2 504	226 623	0.1043
String	100	19 385	19 385	0.0318
GBM	100	650	10 701	0.0566
LGG	32	644	29 374	0.0729
Cerebellum	450	644	30 243	0.0823
TCGA	33	10 459	20 530	0.0501
Movielens 10M	20	71 567	65 133	0.0011
Netflix	20	17 770	480 189	0.0067
Patents	136	10 499	10 511	0.1305

maximum number of iterations without convergence, we bridge the remaining integrality gap simply by applying our proximal operator (see Figure 6.2). To obtain a good reconstruction for Primp, we use a grid width of 0.01. To obtain a binary solution from nmf, we first clamp and then round its factor matrices upon convergence. We set Asso's threshold, gain for covering, and penalty for over-covering each to 1. To achieve a better performance with Asso, we parallelize Asso on 16 CPU cores. Further, because Asso's runtime scales with the number of columns, we reconstruct the *transposed* target whenever it has more columns than rows (see Table 6.1). For example, transposing *GBM*, *LGG*, *Cerebellum*, and *Genomes* is particularly beneficial for Asso, as these datasets have orders of magnitude more columns than rows.

As we can achieve a high similarity with an all-zeros reconstruction (of sparse data), or a perfect recall with an all-ones reconstruction, we also report the *relative loss* (lower is better),

$$\|A\|_1 \|A - U \circ V\|_1 \ ,$$

between the target matrices and their reconstructions.

We show relative loss, similarity, recall, and runtime of Asso, Grecond, Sofa, OrM, Binaps, nmf, Primp, and Elbmf, applied to all real-world datasets, targeting a given matrix rank, in Figure 6.8. The cover-based Grecond and Asso show comparable loss, similarity, and recall.

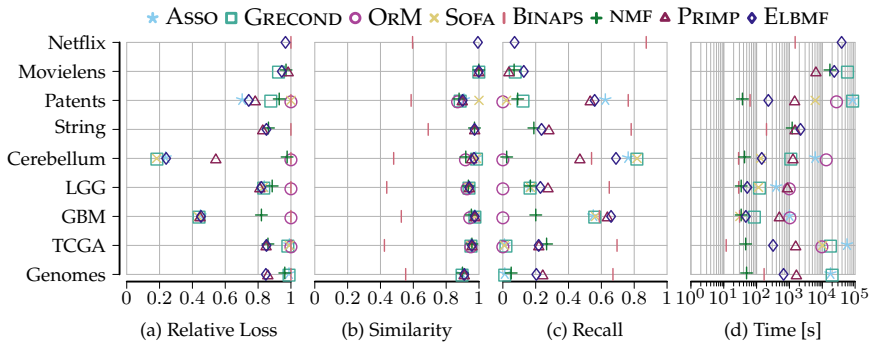
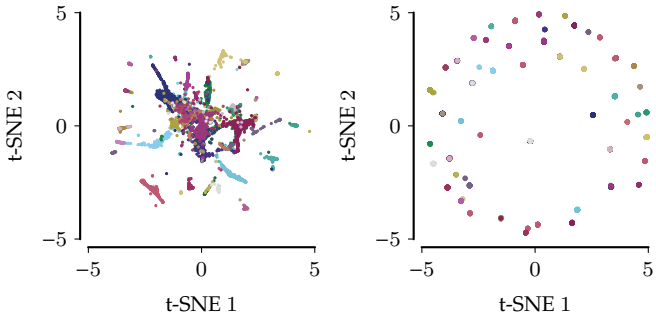


FIGURE 6.8: ELBMF factorizes real-world data with high similarity and recall, as well as low relative loss and runtime. We report *relative loss*, *similarity*, *recall*, and *runtime* for 9 real-world matrices and their reconstructions by ASSO, GRECOND, ORM, SOFA, BINAPS, NMF, PRIMP, and ELBMF.

Both perform better on smaller matrices (*LGG*, *GBM*, or *Cerebellum*) and struggle with complex matrices (e.g., *TCGA* or *Genomes*). On the complex matrices (e.g., *TCGA*, *String*, or *Movielens*), although always outperformed by ELBMF, we see that the *rounded* NMF reconstructions are surprisingly good, occasionally surpassing dedicated BMF methods, such as ASSO, GRECOND, ORM, and SOFA. Across the board, GRECOND, ASSO, ORM, SOFA, BINAPS, and NMF almost always result in considerably higher loss than ELBMF. Compared to the close competitor PRIMP, our method ELBMF always results in lower reconstruction loss. We observe the largest gap between the two on the Cerebellum dataset, where PRIMP's grid-search procedure fails to find suitable thresholds. This is an impressive result because unlike PRIMP, ELBMF does not require heavy post-processing.

In Figure 6.8b, we see that all methods except BINAPS result in a high similarity, which implies they are sparsity-inducing. As BINAPS overfits and densely reconstructs sparse inputs, it surpasses sparsity-inducing methods in recall. Considering non-overfitting methods, however, ELBMF is among the best-performing in terms or recall, often outperforming PRIMP, while under significantly stronger regularization. When PRIMP has a higher recall (e.g., *Genomes*), this often comes with a higher loss than ELBMF. We see in Figure 6.8d (log scale) that—except for a few cases—ASSO, GRECOND, ORM, SOFA, and PRIMP are slower



(a) TCGA (b) Reconstructed TCGA

Figure 6.9: Elbmf discovers hidden structure in gene expression data. We show the two-dimensional t-SNE embedding of the TCGA dataset (a) and the embedding of its reconstruction from Elbmf (b), where each point corresponds to one of the 10 459 patients, colored by cancer type. While the cancer types are hard to differentiate in the embedding of the original dataset (a), they are separated into easily distinguishable clusters in the embedding of our reconstruction (b).

than Elbmf. Although nmf is less constrained than Elbmf, both are almost on par when it comes to runtime. Degraded by post-processing, our closed competitor Primp is almost always much slower than Elbmf, and it struggles with *Netflix*. Only Binaps and Elbmf finished *Netflix*—however, only Elbmf did so at a reasonable loss, considering the given target rank.

6.4.3 Exploration of Gene Expression Data with Elbmf

Knowing that Elbmf performs well quantitatively, we ask whether its outputs are also interpretable. To this end, we take a closer look at the *TCGA* data, which contains the expression levels of 20 530 genes from 10 459 patients, who are labeled with 33 cancer types. Since we are interested in retaining high gene expression levels only, we set expression levels to 1 if their z-scores fall into the top 5% quantile, and to 0 otherwise [103]. We run Elbmf on this dataset, targeting a rank of 33.

To learn whether our method groups patients meaningfully, we visualize the target matrix and its reconstruction. As the target matrix is high-dimensional, we embed both the target matrix and the recon-

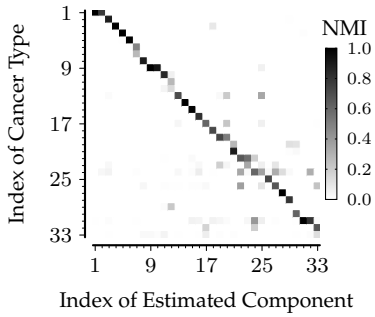


Figure 6.10: Elbmf discovers components that identify cancer types. We show the normalized mutual information between estimated groups and cancer types, and observe that there is an almost 1-to-1 correspondence between our estimated groups and the cancer types.

struction in a two-dimensional space using t-SNE [116] as illustrated in Figure 6.9, where each color corresponds to one cancer type. In Figure 6.9a, we find that when embedding the target matrix directly, the cancer types are highly overlapping and hard to distinguish without the color coding. In contrast, when embedding our reconstruction, depicted in Figure 6.9b, we see a clean segmentation into clusters that predominantly contain a single cancer type.

To better understand these results, we quantify the association between our 33 estimated components and the ground-truth cancer types by computing the *normalized mutual information* matrix, visualized in Figure 6.10. This matrix is noticeably sparse, which leads to a clean segmentation. Upon closer inspection with Enrichr [31], the associations we discover turn out to be biologically meaningful. For example, we find that Elbmf associates a set of 356 genes to patients with *thyroid carcinoma*. This component is associated with *thyroid hormone generation* and *thyroid gland development*, and *statistically significantly* so—even under a strict False Discovery Control, with p-values as low as 2.574×10^{-8} and 6.530×10^{-6}.

6.5 Conclusion

We introduced Elbmf to efficiently factorize Boolean matrices using an elegant and simple algorithm that, unlike its closest competitors,

does not rely on heavy post-processing. Elbmf decomposes Boolean matrices as the Boolean product of two Boolean matrices, obtained via a relaxed optimization scheme using linear algebra of intermediate continuous matrices. It solves this problem by leveraging an efficiently computable proximal operator, derived from the innovative Elb regularizer, and using a regularization rate to obtain Boolean factors upon convergence. Experimentally, we have shown that Elbmf works well in practice. It operates reliably on synthetic data, outperforms the discrete state of the art, is at least as good as the best relaxations on real data, and yields interpretable results even in difficult domains—without relying on post-processing.

Although Elbmf works well overall, it has two bottlenecks. First, by randomly initializing factors, we start with highly dense matrices, thus prohibiting efficient sparse matrix operations. This is not ideal for sparse datasets that are too large to fit into memory, and future research on sparse initialization of *Boolean* factors will benefit not only Elbmf but also many other methods. Second, the larger the datasets, the higher the cost of computing gradients, and future work might adapt stochastic gradient methods for Elbmf to mitigate this problem. Thus, while we have made considerable progress in discovering insightful patterns at scale through our BMF relaxation, there still remains a lot to be discovered.

7
Conclusion

Motivated by real-world needs arising in high-stakes and quality-demanding scientific domains, in this thesis, we developed methods for discovering *insightful patterns*: sets of strongly associated features that are *informative*, *contrasting*, *probabilistically sound*, *statistically sound*, and discoverable using *scalable* algorithms. To conclude this work, in the following, we summarize these methods, discuss their commonalities, and discuss opportunities for future research.

7.1 Retrospective

In Chapter 2, we discussed the maximum entropy distribution in detail, and we introduced an efficient inference strategy for probabilistic models of discrete sets, thus laying the theoretical foundation for all subsequent chapters. Using the maximum entropy distribution, we introduced two algorithms, Disc and Desc, to unveil the pattern composition of a dataset. Disc decomposes a dataset into statistically significantly diverging groups, and Desc efficiently identifies insightful patterns in groups.

In Chapter 3, we considered groups of graphs. We introduced a maximum entropy distribution of paths in graph groups to model subgraph patterns soundly, and proposed the Gragra algorithm to discover contrasting graph patterns with statistical guarantees.

In Chapter 4, we considered insightful patterns for which we could guarantee statistical soundness using hypothesis testing. We introduced the concept of sequentially significant pattern sets, proposed

a novel notion of informativeness using significant unexpectedness, and developed two online false-discovery correction schemes, which we summarized in the Spass framework.

To model patterns probabilistically soundly, Desc, Disc, Gragra, and Spass all use the maximum entropy distribution. As this distribution is either computationally costly or restricted in its expressivity, in Chapter 5, we mitigated this limitation. We proposed a PAC maximum entropy factorization problem and a greedy set-cover algorithm inference solution, thus allowing us to efficiently discover the factorization from data while trading inference complexity with factorization quality.

Although relaxing the maximum entropy distribution could speed up the discovery of pattern sets in binary tabular data, as pattern-set miners rely on combinatorial search, they still struggle to report results on high-dimensional datasets with thousands of features. To obtain insightful patterns that scale to such datasets, in Chapter 6, we combined the interpretability of Boolean matrix factorization (BMF) with the scalability of continuous matrix factorization. Again, relaxing was key: We relaxed BMF as a continuous optimization problem using our novel elastic-binary Elb regularizer, from which we derived a proximal gradient algorithm. Our method, Elbmf, proved to be competitive with the state of the art without relying on the otherwise ubiquitous heavy post-processing. It thus allows us to identify and express groups in data in terms of insightful patterns.

7.2 Commonalities between Chapters

All methods discussed in this thesis are specialized on discrete data, be it binary tabular data or graphs, and most methods are based on the maximum-entropy principle. However, as depicted in Figure 7.1, our methods are also interconnected in many other ways, which we discuss in the following.

The development of Desc in Chapter 2 was inspired by the need to discover characteristics and commonalities between multiple groups in the data, for which we leveraged the maximum entropy distribution,

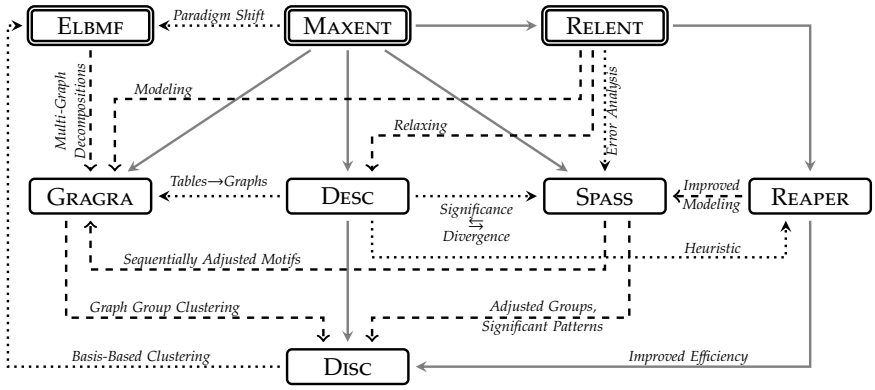


FIGURE 7.1: Web of the science included in this dissertation. We depict *dependencies* (solid lines), *inspirations* (dotted lines), and *applicabilities* (dashed lines) between the research topics covered by this dissertation.

which we built upon to discover statistically significantly diverging groups in the data using DISC. As DISC, however, does not efficiently scale to large datasets, we were motivated to investigate an efficient estimation of the expectation of our maximum entropy distribution. This led us to develop the relaxed maximum entropy distribution in Chapter 5 and the pattern-set miner REAPER, which we then used to not only improve the performance of DISC but also its modeling capabilities. Its intended approximation error, however, might lead to spurious discoveries. To assess the approximation error and incorporate a margin of error into our score, we developed a measure that ultimately resembled Student's t-test. Noticing this, we resorted to well-understood statistics, which brought us to the development of SPASS in Chapter 4. In this chapter, we elucidated the relationship between the p-values used by SPASS and the Kullback-Leibler divergence used by DESC, thus clarifying the link between information-theoretic redundancy and statistical redundancy of patterns. Now, SPASS can serve as a direct replacement of DESC, and it enables a fully statistical-testing-based composition discovery using DISC, which we could additionally improve with our online adjustment strategies.

At the same time, we brought ideas from information-theoretical pattern-set mining like DESC to graph group analysis, which led to GRAGRA, for which we again took inspiration from statistical testing,

thus building a bridge from Desc to graphs. To also close the gap between Spass and Gragra, we can transfer the former's online adjustment strategies to the latter. Although Gragra's distribution can easily be relaxed, it is ultimately maximum-entropy based, which allows us to directly apply Disc to identify statistically significant clusters in a graph collection.

One of the critical challenges faced by pattern-set discovery methods is the computational complexity of searching for the best pattern candidate to be included in the maximum-entropy model. As our search space is too large to be instantiated, we resorted to iterative search, pruning, and relaxations to the distribution. In the worst case, however, those solutions still do not scale to high-dimensional datasets with thousands of features. To overcome this, a paradigm shift in our approach to characteristics and commonalities was needed. For this, we translated the problem from greedy combinatorial algorithms to convex optimization, which led to the development of Elbmf. Use cases of Elbmf go beyond discovering patterns, as it can also be used to identify groups via its basis matrix, as shown in Figure 6.10, which we wish to further improve by transferring the idea of significantly differently distributed groups of Disc to clustering via Elbmf. Although we can also directly apply Elbmf to groups of graphs, this could result in non-connected patterns, which we could prevent by bringing Gragra's concept of connected subgraph patterns to Elbmf.

7.3 Outlook

While this thesis contributes theoretical and methodological groundwork for discovering insightful patterns, it leaves ample room for further research. In addition to the discussions presented in each chapter, we aim to further develop our maximum-entropy distributions by incorporating more expressive first-order logical expressions as patterns. This would enhance the capabilities of our methods, including Desc, Disc, Reaper, Gragra, and Spass. One important special case we aim to explore is directly modeling mutual exclusivity between observations (cf. Fischer et al. [57]), which is currently only modeled

indirectly as probabilistically independent factors. This would greatly improve the expressiveness of our models and their applicability to various fields.

There are, however, other approaches to pattern discovery in tabular Boolean data, including the prominent Boolean matrix factorization and Binary autoencoders. Those have in common that they operate on relaxed constraints, which allow for continuous optimization approaches. Whereas Binaps [55] uses a heuristic pattern extractor on top of a mathematically ad-hoc gradient squashing for autoencoders, Elbmf allows for immediate interpretability upon convergence by using a mathematically well-understood proximal optimization and regularization rate. In the end, both methods serve a similar purpose, which they approach via different avenues. We can use concepts developed in Elbmf to merge those avenues together. That is, to combine the best of both worlds, we can straightforwardly apply Elb's well-understood proximal operator and regularization rate in conjunction with a proximal (stochastic) gradient descent, for training quantized binary autoencoders efficiently, to then appropriate either Fischer et al.'s pattern extraction heuristic [55, 56] or well-understood explainability methods with guarantees. In preliminary experiments exploring just this idea, we saw an improvement over the state of the art, due to our proposed proximal-based optimization schemes.

One important application of Elbmf lies in the biomedical domain, in which privacy concerns often prohibit the compilation of larger shared datasets from multiple sources. We see potential in a distributed learning environment that addresses this concern in terms of *federated* matrix factorization methods that will allow us to discover shared components from private data, thus ensuring stakeholder privacy and security in theory and in practice while enabling scientific discoveries and increasing our understanding of distributed learning.

Inspired by recent developments in explainable machine learning via pattern mining [55], we see future opportunities for Elbmf to dissect neural activations of deep neural networks by identifying groups neurons that are characterized by unique neural network ac-

tivations patterns, thus allowing for correlations of the input with straightforwardly interpretable sets of activation patterns.

Our Elbmf method is well-suited for analyzing adjacency matrices, which are commonly used in network analysis. However, the current version of our method does not take into account the underlying graph semantics of the adjacency matrix, which might result in disconnected subgraph components. By specializing Elbmf to focus on identifying coherent and connected graph motifs, we believe that it would be able to extract more meaningful and relevant information from the data, and thus improve its expressivity in the context of network analysis.

In its causal interpretation, Simpson's paradox [140, 162] describes a potential danger of drawing wrong causal conclusions from grouped data. Future research includes leveraging our tools that decompose data into groups, to aid causal discovery by controlling for potential group-defining confounders. As there cannot be a causal relationship without correlation, a potential prospective investigation is using our statistically sound patterns (correlations) to equip causal discovery with the information necessary for further causal analysis.

On the flip side, our algorithms could also benefit from considering causal dependencies. That is, as science ultimately seeks to understand causal relationships between observations and effects of interest, we would improve the expressivity of our methods by enabling them to discover causal dependencies from data (cf. Marx et al. [121]) in follow-up studies. Our graph group analysis, for example, currently models paths of associated nodes in a network. By restricting our method to only allow causal edges, we would be able to identify causal structure in undirected graphs and thus gain a deeper understanding of the underlying mechanisms.

In addition to the specific directions already discussed toward the end of each chapter or in the above, we see particular potential for future work on a fundamental question:

> How can we integrate methods for automated feature association discovery into scientific workflows?

This question targets the relationship between computer science on the one hand and the domain sciences on the other hand. Its variants occur at all stages of the method-development process:

How can we ensure that the methods we develop to solve *abstract* problems can be translated to tackle the *concrete* problems by which they were motivated? How can we *realistically* evaluate methods that are designed to expand our scientific knowledge beyond what is already known without sacrificing *rigor*? How can we, perhaps even interactively and adaptively, inform our methods about what is already *known*, such that they may focus on revealing what has hitherto been *unknown*? How can we provide trustworthy methods that tell us when their predictions are *uncertain*?

Answering such questions requires close collaboration between computer scientists and domain scientists or real-world experts, which will help us advance toward our goal of making a difference in the real world by putting insightful patterns into practice.

References

[1] Charu C Aggarwal. "An introduction to frequent pattern mining". In: *Frequent pattern mining*. 2014, pp. 1–17 (cit. on p. 14).

[2] Charu C Aggarwal, Mansurul A Bhuiyan, and Mohammad Al Hasan. "Frequent pattern mining algorithms: A survey". In: *Frequent pattern mining*. 2014, pp. 19–64 (cit. on p. 14).

[3] Charu C. Aggarwal and Jiawei Han, eds. *Frequent Pattern Mining*. Springer, 2004 (cit. on p. 98).

[4] Gaël Aglin, Siegfried Nijssen, and Pierre Schaus. "Learning Optimal Decision Trees Using Caching Branch-and-Bound Search". In: *AAAI* 34 (2020), pp. 3146–3153 (cit. on p. 108).

[5] Rakesh Agrawal and Ramakrishnan Srikant. "Fast Algorithms for Mining Association Rules". In: *VLDB*. Vol. 1215. 1994, pp. 487–499 (cit. on pp. 14, 47, 98, 134).

[6] Ehud Aharoni and Saharon Rosset. "Generalized α-investing: definitions, optimality results and application to public databases". In: *J. R. Stat. Soc. B* 76.4 (2013), pp. 771–794 (cit. on p. 95).

[7] H. Akaike. "A new look at the statistical model identification". In: *IEEE Transactions on Automatic Control* 19.6 (1974), pp. 716–723 (cit. on pp. 46, 164).

[8] Lucia Alessi, Matteo Barigozzi, and Marco Capasso. "Improved penalization for determining the number of factors in approximate factor models". In: *Statistics & Probability Letters* 80.23-24 (2010), pp. 1806–1813 (cit. on pp. 46, 164).

[9] Miguel Araujo, Pedro Manuel Pinto Ribeiro, and Christos Faloutsos. "FastStep: Scalable Boolean Matrix Decomposition". In: *Advances in Knowledge Discovery and Data Mining - 20th Pacific-Asia Conference, PAKDD 2016, Auckland, New Zealand, April 19-22, 2016, Proceedings, Part I*. Vol. 9651. 2016, pp. 461–473 (cit. on p. 157).

[10] Alexander Bagaev et al. "Conserved pan-cancer microenvironment subtypes predict response to immunotherapy". In: *Cancer cell* 39.6 (2021), pp. 845–865 (cit. on p. 13).

[11] Jushan Bai and Serena Ng. "Determining the number of factors in approximate factor models". In: *Econometrica* 70.1 (2002), pp. 191–221 (cit. on pp. 46, 164).

[12] Matthew H Bailey et al. "Comprehensive characterization of cancer driver genes and mutations". In: *Cell* 173.2 (2018), pp. 371–385 (cit. on p. 13).

[13] Jörg Balss et al. "Analysis of the IDH1 codon 132 mutation in brain tumors". In: *Acta neuropathologica* 116.6 (2008), pp. 597–602 (cit. on p. 26).

[14] Deanna M Barch et al. "Function in the human connectome: task-fMRI and individual differences in behavior". In: *Neuroimage* 80 (2013), pp. 169–189 (cit. on p. 13).

[15] Andrew R. Barron and Chyong-Hwa Sheu. "Approximation of Density Functions by Sequences of Exponential Families". In: *The Annals of Statistics* 19.3 (1991) (cit. on p. 133).

[16] Danielle S Bassett and Olaf Sporns. "Network neuroscience". In: *Nature Neuroscience* 20.3 (2017), pp. 353–364 (cit. on p. 76).

[17] Stephen D. Bay and Michael J. Pazzani. "Detecting Group Differences: Mining Contrast Sets". In: *Data Min. Knowl. Discov.* 5.3 (2001), pp. 213–246 (cit. on pp. 93, 99).

[18] N. Bebiano, J. da Providência, and J. P. da Providência. "Toward non-Hermitian quantum statistical thermodynamics". In: *Journal of Mathematical Physics* 61.2 (2020), p. 022102 (cit. on p. 29).

[19] Radim Belohlávek and Martin Trnecka. "A new algorithm for Boolean matrix factorization which admits overcovering". In: *Discret. Appl. Math.* 249 (2018), pp. 36–52 (cit. on pp. 147, 155).

[20] Radim Belohlávek and Vilém Vychodil. "Discovery of optimal factors in binary data via a novel method of matrix decomposition". In: *J. Comput. Syst. Sci.* 76.1 (2010), pp. 3–20 (cit. on pp. 147, 155, 157).

[21] Yoav Benjamini and Yosef Hochberg. "Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing". In: *J Royal Stat. S B (Methodological)* 57.1 (1995), pp. 289–300 (cit. on p. 94).

[22] Tijl De Bie. "Maximum entropy models and subjective interestingness: an application to tiles in binary databases". In: *Data Min. Knowl. Discov.* 23.3 (2011), pp. 407–446 (cit. on pp. 29, 116).

[23] Claudio Bierig and Alexey Chernov. "Approximation of probability density functions by the Multilevel Monte Carlo Maximum Entropy method". In: *J. Comput. Phys.* 314 (2016), pp. 661–681 (cit. on p. 133).

[24] Jérôme Bolte, Shoham Sabach, and Marc Teboulle. "Proximal alternating linearized minimization for nonconvex and nonsmooth problems". In: *Math. Program.* 146.1-2 (2014), pp. 459–494 (cit. on p. 151).

[25] C. E. Bonferroni. "Teoria Statistica Delle Classi e Calcolo Delle Probabilita". In: *Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commericiali di Firenze* 8 (1936), pp. 3–62 (cit. on pp. 43, 93, 99).

[26] L. Breiman et al. "Classification and Regression Trees". In: (1983) (cit. on p. 107).

[27] Kailash Budhathoki and Jilles Vreeken. "The Difference and the Norm - Characterising Similarities and Differences Between Databases". In: *ECML PKDD*. Vol. 9285. 2015, pp. 206–223 (cit. on pp. 47, 48, 98).

[28] Ed Bullmore and Olaf Sporns. "Complex brain networks: graph theoretical analysis of structural and functional systems". In: *Nature Reviews Neuroscience* 10.3 (2009), pp. 186–198 (cit. on p. 76).

[29] Bureau of Transportation Statistics. *Data Bank 28DS - T-100 Domestic Segment Data (World Area Code)*. https://www.bts.gov/browse-statistical-products-and-data/bts-publications/data-bank-28ds-t-100-domestic-segment-data. 2021 (cit. on p. 78).

[30] Deepayan Chakrabarti et al. “Fully automatic cross-associations”. In: *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22-25, 2004*. 2004, pp. 79–88 (cit. on p. 46).

[31] Edward Y. Chen et al. “Enrichr: interactive and collaborative HTML5 gene list enrichment analysis tool”. In: *BMC Bioinform.* 14 (2013), p. 128 (cit. on p. 169).

[32] Junxiang Chen et al. “Interpretable Clustering via Discriminative Rectangle Mixture Model”. In: *IEEE 16th International Conference on Data Mining, ICDM 2016, December 12-15, 2016, Barcelona, Spain*. 2016, pp. 823–828 (cit. on p. 47).

[33] Herman Chernoff. “A Measure of Asymptotic Efficiency for Tests of a Hypothesis Based on the Sum of Observations”. In: *Ann Math Stat* 23.4 (1952), pp. 493–507 (cit. on p. 91).

[34] Martino Ciaperoni, Han Xiao, and Aristides Gionis. “Concise and interpretable multi-label rule sets”. In: *IEEE International Conference on Data Mining, ICDM 2022, Orlando, FL, USA, November 28 - Dec. 1, 2022*. 2022, pp. 71–80 (cit. on p. 47).

[35] Corinna Coupette, Sebastian Dalleiger, and Jilles Vreeken. “Differentially Describing Groups of Graphs”. In: *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*. 2022, pp. 3959–3967 (cit. on pp. 23, 59).

[36] Corinna Coupette and Jilles Vreeken. “Graph Similarity Description: How Are These Graphs Similar?” In: *KDD ’21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*. 2021, pp. 185–195 (cit. on p. 68).

[37] Thomas M. Cover and Joy A. Thomas. “Elements of Information Theory”. In: (2005) (cit. on p. 126).

[38] Cameron Craddock et al. “The neuro bureau preprocessing initiative: open sharing of preprocessed neuroimaging data and derivatives”. In: *Frontiers in Neuroinformatics* 7 (2013) (cit. on p. 76).

[39] I. Csiszar. "*I*-Divergence Geometry of Probability Distributions and Minimization Problems". In: *The Annals of Probability* 3.1 (1975), pp. 146–158 (cit. on pp. 30, 116, 127, 132).

[40] Christina Curtis and et.al. "The genomic and transcriptomic architecture of 2,000 breast tumours reveals novel subgroups". In: *Nature* 486.7403 (2012), pp. 346–352 (cit. on p. 110).

[41] Sebastian Dalleiger and Jilles Vreeken. "Discovering Significant Patterns under Sequential False Discovery Control". In: *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*. 2022, pp. 263–272 (cit. on pp. 23, 85, 145).

[42] Sebastian Dalleiger and Jilles Vreeken. "Efficiently Factorizing Boolean Matrices using Proximal Gradient Descent". In: *Thirty-Sixth Conference on Neural Information Processing Systems (NeurIPS)*. 2022, pp. 4736–4748 (cit. on pp. 23, 46).

[43] Sebastian Dalleiger and Jilles Vreeken. "Explainable Data Decompositions". In: *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. 2020, pp. 3709–3716 (cit. on pp. 22, 25, 68, 98, 99, 106, 116, 120, 131, 134, 140–142).

[44] Sebastian Dalleiger and Jilles Vreeken. "The Relaxed Maximum Entropy Distribution and its Application to Pattern Discovery". In: *20th IEEE International Conference on Data Mining, ICDM 2020, Sorrento, Italy, November 17-20, 2020*. 2020, pp. 978–983 (cit. on pp. 23, 68, 115).

[45] J. N. Darroch and D. Ratcliff. "Generalized Iterative Scaling for Log-Linear Models". In: *The Annals of Mathematical Statistics* 43.5 (1972), pp. 1470–1480 (cit. on pp. 30, 116).

[46] A. P. Dempster, N. M. Laird, and D. B. Rubin. "Maximum Likelihood from Incomplete Data Via theEMAlgorithm". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 39.1 (1977), pp. 1–22 (cit. on p. 46).

[47] Prabhjot S Dhadialla et al. "Maximum-entropy network analysis reveals a role for tumor necrosis factor in peripheral nerve development and function". In: *Proceedings of the National Academy of Sciences* 106.30 (2009), pp. 12494–12499 (cit. on p. 29).

[48] Inderjit S. Dhillon, Subramanyam Mallela, and Dharmendra S. Modha. "Information-theoretic co-clustering". In: *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 24 - 27, 2003*. 2003, pp. 89–98 (cit. on p. 46).

[49] Purushottam D. Dixit. "A maximum entropy thermodynamics of small systems". In: *The Journal of Chemical Physics* 138.18 (2013), p. 184111 (cit. on p. 29).

[50] Miroslav Dudík, Steven J. Phillips, and Robert E. Schapire. "Maximum Entropy Density Estimation with Generalized Regularization and an Application to Species Distribution Modeling". In: *J. Mach. Learn. Res.* 8 (2007), pp. 1217–1260 (cit. on p. 133).

[51] Daniele Durante, David B. Dunson, and Joshua T. Vogelstein. "Nonparametric Bayes Modeling of Populations of Networks". In: *Journal of the American Statistical Association* 112.520 (2017), pp. 1516–1530 (cit. on p. 69).

[52] Daniele Durante, David B Dunson, et al. "Bayesian inference and testing of group differences in brain networks". In: *Bayesian Analysis* 13.1 (2018), pp. 29–58 (cit. on p. 69).

[53] Martin Ester et al. "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise". In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA*. 1996, pp. 226–231 (cit. on p. 50).

[54] Giorgio Fagiolo, Javier Reyes, and Stefano Schiavo. "The evolution of the world trade web: a weighted-network analysis". In: *Journal of Evolutionary Economics* 20.4 (2010), pp. 479–514 (cit. on p. 81).

[55] Jonas Fischer, Anna Oláh, and Jilles Vreeken. "What's in the Box? Exploring the Inner Life of Neural Networks with Robust Rules". In: *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*. Vol. 139. 2021, pp. 3352–3362 (cit. on p. 175).

[56] Jonas Fischer and Jilles Vreeken. "Differentiable Pattern Set Mining". In: *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*. 2021, pp. 383–392 (cit. on pp. 157, 165, 175).

[57] Jonas Fischer and Jilles Vreeken. "Discovering Succinct Pattern Sets Expressing Co-Occurrence and Mutual Exclusivity". In: *KDD '20*. 2020, pp. 813–823 (cit. on pp. 98, 174).

[58] Alex Fornito, Andrew Zalesky, and Michael Breakspear. "The connectomics of brain disorders". In: *Nature Reviews Neuroscience* 16.3 (2015), pp. 159–172 (cit. on p. 76).

[59] Jaroslav M. Fowkes and Charles Sutton. "A Bayesian Network Model for Interesting Itemsets". In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2016, Riva del Garda, Italy, September 19-23, 2016, Proceedings, Part II*. Vol. 9852. 2016, pp. 410–425 (cit. on p. 47).

[60] Jaroslav M. Fowkes and Charles Sutton. "A Bayesian Network Model for Interesting Itemsets". In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2016, Riva del Garda, Italy, September 19-23, 2016, Proceedings, Part II*. Vol. 9852. 2016, pp. 410–425 (cit. on pp. 98, 141).

[61] Cristian A Gallo et al. "Discretization of gene expression data revised". In: *Briefings in bioinformatics* 17.5 (2016), pp. 758–770 (cit. on p. 102).

[62] Debarghya Ghoshdastidar et al. "Two-sample hypothesis testing for inhomogeneous random graphs". In: *AnnalsStatistics* 48.4 (2020), pp. 2208–2229 (cit. on p. 69).

[63] Cedric E Ginestet et al. "Hypothesis testing for network data in functional neuroimaging". In: *AnnalsAppliedStatistics* 11.2 (2017), pp. 725–750 (cit. on p. 69).

[64] Gene H. Golub and Charles F. Van Loan. *Matrix Computations, Third Edition*. Johns Hopkins University Press, 1996 (cit. on p. 146).

[65] Guilherme Gomes, Vinayak A. Rao, and Jennifer Neville. "Multilevel Hypothesis Testing for Populations of Heterogeneous Networks". In: *IEEE International Conference on Data Mining, ICDM 2018, Singapore, November 17-20, 2018*. 2018, pp. 977–982 (cit. on p. 69).

[66] Peter D. Grünwald. *The Minimum Description Length Principle*. The MIT Press, 2007 (cit. on p. 164).

[67] Suriya Gunasekar et al. “Implicit Regularization in Matrix Factorization”. In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. 2017, pp. 6151–6159 (cit. on pp. 46, 164).

[68] Tias Guns, Siegfried Nijssen, and Luc De Raedt. “k-Pattern set mining under constraints”. In: *IEEE Transactions on Knowledge and Data Engineering* 25.2 (2011), pp. 402–418 (cit. on p. 14).

[69] Ali Haddad et al. “Identifying Dynamics of Brain Function Via Boolean Matrix Factorization”. In: *52nd Asilomar Conference on Signals, Systems, and Computers, ACSSC 2018, Pacific Grove, CA, USA, October 28-31, 2018*. 2018, pp. 661–665 (cit. on p. 146).

[70] Wilhelmiina Hämäläinen. “Kingfisher: An Efficient Algorithm for Searching for Both Positive and Negative Dependency Rules with Statistical Significance Measures”. In: *Knowl Inf Syst* 32.2 (2012), pp. 383–414 (cit. on p. 98).

[71] Jiawei Han et al. “Frequent pattern mining: current status and future directions”. In: *Data mining and knowledge discovery* 15.1 (2007), pp. 55–86 (cit. on p. 14).

[72] Ye He, Lisa Byrge, and Daniel P Kennedy. “Nonreplication of functional connectivity differences in autism spectrum disorder across multiple sites and denoising strategies”. In: *Human Brain Mapping* 41.5 (2020), pp. 1334–1350 (cit. on p. 76).

[73] Sibylle Hess and Katharina Morik. “C-SALT: Mining Class-Specific ALTerations in Boolean Matrix Factorization”. In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18-22 , 2017, Proceedings, Part I*. Vol. 10534. 2017, pp. 547–563 (cit. on p. 157).

[74] Sibylle Hess, Katharina Morik, and Nico Piatkowski. “The PRIMPING routine - Tiling through proximal alternating linearized minimization”. In: *Data Min. Knowl. Discov.* 31.4 (2017), pp. 1090–1131 (cit. on pp. 147, 149, 157).

[75] Sibylle Hess, Nico Piatkowski, and Katharina Morik. “The Trustworthy Pal: Controlling the False Discovery Rate in Boolean Matrix Factorization”. In: *Proceedings of the 2018 SIAM International Conference on Data Mining, SDM 2018, May 3-5, 2018, San Diego Marriott Mission Valley, San Diego, CA, USA*. 2018, pp. 405–413 (cit. on p. 147).

[76] Sibylle Hess et al. “BROCCOLI: overlapping and outlier-robust biclustering through proximal stochastic gradient descent”. In: *Data Min. Knowl. Discov.* 35.6 (2021), pp. 2542–2576 (cit. on pp. 147, 157).

[77] Martijn P van den Heuvel and Olaf Sporns. “A cross-disorder connectome landscape of brain dysconnectivity”. In: *Nature Reviews Neuroscience* 20.7 (2019), pp. 435–446 (cit. on p. 76).

[78] Seok-Jun Hong et al. “Toward neurosubtypes in autism”. In: *Biological Psychiatry* 88.1 (2020), pp. 111–128 (cit. on p. 78).

[79] Jocelyn V Hull et al. “Resting-state functional connectivity in autism spectrum disorders: a review”. In: *Frontiers in Psychiatry* 7 (2017), p. 205 (cit. on p. 61).

[80] Dmitry I. Ignatov et al. “Boolean Matrix Factorisation for Collaborative Filtering: An FCA-Based Approach”. In: *Artificial Intelligence: Methodology, Systems, and Applications - 16th International Conference, AIMSA 2014, Varna, Bulgaria, September 11-13, 2014. Proceedings*. Vol. 8722. 2014, pp. 47–58 (cit. on p. 146).

[81] Yu Ito, Shinichi Oeda, and Kenji Yamanishi. “Rank Selection for Non-negative Matrix Factorization with Normalized Maximum Likelihood Coding”. In: *Proceedings of the 2016 SIAM International Conference on Data Mining, Miami, Florida, USA, May 5-7, 2016*. 2016, pp. 720–728 (cit. on pp. 46, 164).

[82] Martin Jaggi and Marek Sulovský. “A Simple Algorithm for Nuclear Norm Regularized Problems”. In: *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*. 2010, pp. 471–478 (cit. on pp. 46, 164).

[83] Adel Javanmard and Andrea Montanari. “On Online Control of False Discovery Rate”. In: *CoRR* (2015) (cit. on p. 112).

[84] Adel Javanmard and Andrea Montanari. “Online rules for control of false discovery rate and false discovery exceedance”. In: *Ann. Statist.* 46.2 (2018), pp. 526 –554 (cit. on pp. 95, 96, 112).

[85] E. T. Jaynes. “Information Theory and Statistical Mechanics”. In: *Physical Review* 106.4 (1957), pp. 620–630 (cit. on pp. 15, 28, 132).

[86] Edwin T Jaynes. “On the rationale of maximum-entropy methods”. In: *PIEEE* 70.9 (1982), pp. 939–952 (cit. on pp. 26, 28, 30, 132).

[87] Bo Kang et al. “SICA: subjectively interesting component analysis”. In: *Data Min. Knowl. Discov.* 32.4 (2018), pp. 949–987 (cit. on p. 116).

[88] Tobias Kaufmann et al. “Delayed stabilization and individualization in connectome development are related to psychiatric disorders”. In: *Nature neuroscience* 20.4 (2017), pp. 513–515 (cit. on p. 13).

[89] Been Kim, Julie A. Shah, and Finale Doshi-Velez. “Mind the Gap: A Generative Approach to Interpretable Feature Selection and Extraction”. In: *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*. 2015, pp. 2260–2268 (cit. on p. 47).

[90] Jace B King et al. “Generalizability and reproducibility of functional connectivity in autism”. In: *Molecular Autism* 10.1 (2019), pp. 1–23 (cit. on p. 76).

[91] Kleanthis-Nikolaos Kontonasios, Jilles Vreeken, and Tijl De Bie. “Maximum entropy models for iteratively identifying subjectively interesting structure in real-valued data”. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2013, pp. 256–271 (cit. on p. 29).

[92] Kleanthis-Nikolaos Kontonasios, Jilles Vreeken, and Tijl De Bie. “Maximum entropy modelling for assessing results on real-valued data”. In: *2011 IEEE 11th International Conference on Data Mining*. IEEE. 2011, pp. 350–359 (cit. on p. 29).

[93] Bernhard Korte and Jens Vygen. “Combinatorial Optimization”. In: *Algorithms and Combinatorics* (2018) (cit. on p. 125).

[94] Andreas Krause and Daniel Golovin. “Submodular Function Maximization”. In: *Tractability: Practical Approaches to Hard Problems*. 2014, pp. 71–104 (cit. on p. 37).

[95] S. Kullback and R. A. Leibler. “On Information and Sufficiency”. In: *The Annals of Mathematical Statistics* 22.1 (1951), pp. 79–86 (cit. on p. 126).

[96] Suprateek Kundu et al. “Integrative learning for population of dynamic networks with covariates”. In: *NeuroImage* 236 (2021), p. 118181 (cit. on p. 69).

[97] Himabindu Lakkaraju, Stephen H. Bach, and Jure Leskovec. "Interpretable Decision Sets: A Joint Framework for Description and Prediction". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*. 2016, pp. 1675–1684 (cit. on p. 47).

[98] Tommaso Lanciano, Francesco Bonchi, and Aristides Gionis. "Explainable Classification of Brain Networks via Contrast Subgraphs". In: *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*. 2020, pp. 3308–3318 (cit. on pp. 69, 70, 76).

[99] Daniel D. Lee and H. Sebastian Seung. "Algorithms for Non-negative Matrix Factorization". In: *Advances in Neural Information Processing Systems 13, Papers from Neural Information Processing Systems (NIPS) 2000, Denver, CO, USA*. 2000, pp. 556–562 (cit. on pp. 146, 157).

[100] Daniel D. Lee and H. Sebastian Seung. "Learning the parts of objects by non-negative matrix factorization". In: *Nature* 401.6755 (1999), pp. 788–791 (cit. on pp. 146, 157).

[101] John Boaz Lee, Ryan A. Rossi, and Xiangnan Kong. "Graph Classification using Structural Attention". In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*. 2018, pp. 1666–1674 (cit. on p. 61).

[102] BCL Lehmann et al. "Characterising group-level brain connectivity: a framework using Bayesian exponential random graph models". In: *NeuroImage* 225 (2021), p. 117480 (cit. on p. 69).

[103] Lifan Liang, Kunju Zhu, and Songjian Lu. "BEM: Mining Coregulation Patterns in Transcriptomics via Boolean Matrix Factorization". In: *Bioinform.* 36.13 (2020), pp. 4030–4037 (cit. on pp. 146, 157, 168).

[104] Hu Liu, Sheng Jin, and Changshui Zhang. "Connectionist Temporal Classification with Maximum Entropy Regularization". In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. 2018, pp. 839–849 (cit. on p. 116).

[105] Yike Liu et al. “Graph Summarization Methods and Applications: A Survey”. In: *ACM Comput. Surv.* 51.3 (2018), 62:1–62:34 (cit. on p. 61).

[106] Felipe Llinares-López et al. “CASMAP: Detection of Statistically Significant Combinations of SNPs in Association Mapping”. In: *Bioinformatics* 35.15 (2019), pp. 2680–2682 (cit. on pp. 86, 98).

[107] Felipe Llinares-López et al. “Fast and memory-efficient significant pattern mining via permutation testing”. In: *KDD15*. 2015, pp. 725–734 (cit. on p. 69).

[108] Felipe Llinares-López et al. “Fast and Memory-Efficient Significant Pattern Mining via Permutation Testing”. In: *KDD*. 2015, pp. 725–734 (cit. on pp. 86, 98, 99).

[109] Felipe Llinares-López et al. “Genome-Wide Genetic Heterogeneity Discovery with Categorical Covariates”. In: *Bioinformatics* 33.12 (2017), pp. 1820–1828 (cit. on pp. 86, 98).

[110] Paulett Lloyd, Matthew C Mahutga, and Jan De Leeuw. “Looking back and forging ahead: Thirty years of social network research on the world-system”. In: *Journal of World-Systems Research* (2009), pp. 48–85 (cit. on p. 81).

[111] Ilenia Lovato et al. “Model-free two-sample test for network-valued data”. In: *Comput. Stat. Data Anal.* 144 (2020), p. 106896 (cit. on p. 69).

[112] Ilenia Lovato et al. “Multiscale null hypothesis testing for network-valued data: Analysis of brain networks of patients with autism”. In: *JRStatistSocC* 70.2 (2021), pp. 372–397 (cit. on p. 69).

[113] Joshua Lukemire et al. “Bayesian joint modeling of multiple brain functional networks”. In: *JASA* (2020), pp. 1–13 (cit. on p. 69).

[114] Simón Lunagómez, Sofia C. Olhede, and Patrick J. Wolfe. “Modeling Network Populations via Graph Distances”. In: *Journal of the American Statistical Association* 116.536 (2021), pp. 2023–2040 (cit. on p. 69).

[115] Andrew L. Maas et al. “Learning Word Vectors for Sentiment Analysis”. In: *AMACL*. 2011, pp. 142–150 (cit. on p. 109).

[116] Laurens Van der Maaten and Geoffrey Hinton. “Visualizing data using t-SNE.” In: *Journal of machine learning research* 9.11 (2008) (cit. on p. 169).

[117] James MacQueen. “Some methods for classification and analysis of multivariate observations”. In: *BSMSP*. 1967, Vol. I: Statistics, pp. 281–297 (cit. on p. 46).

[118] Michael Mampaey, Jilles Vreeken, and Nikolaj Tatti. “Summarizing data succinctly with the most informative itemsets”. In: *ACM Trans. Knowl. Discov. Data* 6.4 (2012), 16:1–16:42 (cit. on pp. 31, 32, 35, 36, 42, 47, 48, 116, 119, 120, 134, 141).

[119] Michael Mampaey, Jilles Vreeken, and Nikolaj Tatti. “Summarizing Data Succinctly with the Most Informative Itemsets”. In: *TKDD* 6.4 (2012), p. 16 (cit. on pp. 99, 106).

[120] Debora S Marks et al. “Protein 3D structure computed from evolutionary sequence variation”. In: *PloS one* 6.12 (2011), e28766 (cit. on p. 29).

[121] Alexander Marx and Jilles Vreeken. “Formally Justifying MDL-based Inference of Cause and Effect”. In: *AAAI Workshop on Information-Theoretic Causal Inference and Discovery (ITCI'22)*. 2022 (cit. on p. 176).

[122] P-AG Maugis et al. “Testing for equivalence of network distribution using subgraph counts”. In: *Journal of Computational and Graphical Statistics* 29.3 (2020), pp. 455–465 (cit. on p. 69).

[123] Gaurav Mendiratta et al. “Cancer gene mutation frequencies for the US population”. In: *Nature communications* 12.1 (2021), pp. 1–11 (cit. on p. 13).

[124] M.L. Menéndez et al. “The Jensen-Shannon divergence”. In: *Journal of the Franklin Institute* 334.2 (1997), pp. 307–318 (cit. on p. 43).

[125] Pauli Miettinen and Stefan Neumann. “Recent Developments in Boolean Matrix Factorization”. In: *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*. 2020, pp. 4922–4928 (cit. on pp. 46, 147, 148, 157).

[126] Pauli Miettinen and Jilles Vreeken. “MDL4BMF: Minimum Description Length for Boolean Matrix Factorization”. In: *ACM Trans. Knowl. Discov. Data* 8.4 (2014), 18:1–18:31 (cit. on pp. 155, 164).

[127] Pauli Miettinen et al. “The Discrete Basis Problem”. In: *IEEE Trans. Knowl. Data Eng.* 20.10 (2008), pp. 1348–1362 (cit. on pp. 46, 146, 147, 155, 157).

[128] Shin-ichi Minato et al. “A Fast Method of Statistical Assessment for Combinatorial Hypotheses Based on Frequent Itemset Enumeration”. In: *Mach Learn. Know Disc. Data*. 2014, pp. 422–436 (cit. on p. 93).

[129] Sylvia D Monson, Norman J Pullman, and Rolf Rees. “A survey of clique and biclique coverings and factorizations of (0, 1)-matrices”. In: *Bull. Inst. Combin. Appl* 14 (1995), pp. 17–86 (cit. on p. 155).

[130] Soumendu Sundar Mukherjee, Purnamrita Sarkar, and Lizhen Lin. “On clustering network-valued data”. In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. 2017, pp. 7071–7081 (cit. on pp. 61, 69).

[131] Stefan Neumann. “Bipartite Stochastic Block Models with Tiny Clusters”. In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. 2018, pp. 3871–3881 (cit. on pp. 147, 157).

[132] Stefan Neumann and Pauli Miettinen. “Biclustering and Boolean Matrix Factorization in Data Streams”. In: *Proc. VLDB Endow.* 13.10 (2020), pp. 1709–1722 (cit. on pp. 46, 147, 157).

[133] Siegfried Nijssen, Tias Guns, and Luc De Raedt. “Correlated itemset mining in ROC space: a constraint programming approach”. In: *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, June 28 - July 1, 2009*. 2009, pp. 647–656 (cit. on pp. 47, 134).

[134] Jason S Nomi and Lucina Q Uddin. “Developmental changes in large-scale network connectivity in autism”. In: *NeuroImage: Clinical* 7 (2015), pp. 732–741 (cit. on p. 76).

[135] James Orlin. “Contentment in graph theory: Covering graphs with cliques”. In: *Indagationes Mathematicae (Proceedings)* 80.5 (1977), pp. 406–424 (cit. on pp. 46, 147).

[136] Pentti Paatero and Unto Tapper. “Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values”. In: *Environmetrics* 5.2 (1994), pp. 111–126 (cit. on pp. 146, 148, 157).

[137] Thomas Pabst et al. “Heterogeneity within AML with CEBPA mutations; only CEBPA double mutations, but not single CEBPA mutations are associated with favourable prognosis”. In: *British journal of cancer* 100.8 (2009), pp. 1343–1346 (cit. on p. 26).

[138] Laetitia Papaxanthos et al. “Finding Significant Combinations of Features in the Presence of Categorical Covariates”. In: *NeurIPS*. 2016, pp. 2279–2287 (cit. on p. 98).

[139] Neal Parikh and Stephen P. Boyd. “Proximal Algorithms”. In: *Found. Trends Optim.* 1.3 (2014), pp. 127–239 (cit. on p. 151).

[140] Judea Pearl and Dana Mackenzie. *The book of why: the new science of cause and effect*. Penguin books, 2018 (cit. on p. 176).

[141] Leonardo Pellegrina, Matteo Riondato, and Fabio Vandin. “SPuManTE: Significant Pattern Mining with Unconditional Testing”. In: *KDD*. 2019, pp. 1528–1538 (cit. on pp. 86, 98, 99).

[142] Leonardo Pellegrina and Fabio Vandin. “Efficient Mining of the Most Significant Patterns with Permutation Testing”. In: *KDD*. 2018, pp. 2070–2079 (cit. on p. 98).

[143] A Petitjean et al. “TP53 mutations in human cancers: functional selection and impact on cancer prognosis and outcomes”. In: *Oncogene* 26.15 (2007), pp. 2157–2165 (cit. on p. 26).

[144] Thomas Pock and Shoham Sabach. “Inertial Proximal Alternating Linearized Minimization (iPALM) for Nonconvex and Nonsmooth Problems”. In: *SIAM J. Imaging Sci.* 9.4 (2016), pp. 1756–1787 (cit. on pp. 151, 156).

[145] Luc De Raedt and Albrecht Zimmermann. “Constraint-based pattern set mining”. In: *proceedings of the 2007 SIAM International conference on Data Mining*. SIAM. 2007, pp. 237–248 (cit. on p. 14).

[146] Aaditya Ramdas et al. “SAFFRON: an Adaptive Algorithm for Online Control of the False Discovery Rate”. In: *ICML*. Vol. 80. 2018, pp. 4283–4291 (cit. on p. 112).

[147] Michael Rapp et al. “Gradient-Based Label Binning in Multi-label Classification”. In: *Machine Learning and Knowledge Discovery in Databases. Research Track - European Conference, ECML PKDD 2021, Bilbao, Spain, September 13-17, 2021, Proceedings, Part III*. Vol. 12977. 2021, pp. 462–477 (cit. on p. 47).

[148] Michael Rapp et al. “Learning Gradient Boosted Multi-label Classification Rules”. In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2020, Ghent, Belgium, September 14-18, 2020, Proceedings, Part III*. Vol. 12459. 2020, pp. 124–140 (cit. on p. 47).

[149] Siamak Ravanbakhsh, Barnabás Póczos, and Russell Greiner. “Boolean Matrix Factorization and Noisy Completion via Message Passing”. In: *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*. Vol. 48. 2016, pp. 945–954 (cit. on p. 157).

[150] Raissa T. Relator, Aika Terada, and Jun Sese. “Identifying Statistically Significant Combinatorial Markers for Survival Analysis”. In: *BMC Med. Genomics* 11.2 (2018), p. 31 (cit. on pp. 86, 98).

[151] Mark E Robson. “Clinical considerations in the management of individuals at risk for hereditary breast and ovarian cancer”. In: *Cancer Control* 9.6 (2002), pp. 457–465 (cit. on p. 26).

[152] Edmund T. Rolls et al. “Automated anatomical labelling atlas 3”. In: *NeuroImage* 206 (2020) (cit. on p. 76).

[153] Tammo Rukat, Christopher C. Holmes, and Christopher Yau. “Probabilistic Boolean Tensor Decomposition”. In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*. Vol. 80. 2018, pp. 4410–4419 (cit. on pp. 147, 157).

[154] Tammo Rukat, Dustin Lange, and Cedric Archambeau. “An interpretable latent variable model for attribute applicability in the Amazon catalogue”. In: *NeurIPS 2017*. 2017 (cit. on pp. 147, 157).

[155] Tammo Rukat et al. “Bayesian Boolean Matrix Factorisation”. In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*. Vol. 70. 2017, pp. 2969–2978 (cit. on pp. 147, 157).

[156] Michael Alan Sacks, Marc J Ventresca, and Brian Uzzi. “Global institutions and networks: Contingent change in the structure of world trade advantage, 1965-1980”. In: *American Behavioral Scientist* 44.10 (2001), pp. 1579–1601 (cit. on p. 81).

[157] Ellis Scharfenaker and Jangho Yang. "Maximum entropy economics: where do we stand?" In: *The European Physical Journal Special Topics* 229.9 (2020), pp. 1573–1575 (cit. on p. 29).

[158] Gideon Schwarz. "Estimating the dimension of a model". In: *AnnalsStatistics* (1978), pp. 461–464 (cit. on pp. 33, 63).

[159] Neil Shah et al. "TimeCrunch: Interpretable Dynamic Graph Summarization". In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*. 2015, pp. 1055–1064 (cit. on p. 69).

[160] Claude Elwood Shannon. "A mathematical theory of communication". In: *The Bell system technical journal* 27.3 (1948), pp. 379–423 (cit. on p. 15).

[161] Mirko Signorelli and Ernst C Wit. "Model-based clustering for populations of networks". In: *Statistical Modelling* 20.1 (2020), pp. 9–29 (cit. on p. 69).

[162] Edward H Simpson. "The interpretation of interaction in contingency tables". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 13.2 (1951), pp. 238–241 (cit. on p. 176).

[163] Mohit Singh and Nisheeth K. Vishnoi. "Entropy, optimization and counting". In: *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*. 2014, pp. 50–59 (cit. on p. 133).

[164] Koen Smets and Jilles Vreeken. "Slim: Directly Mining Descriptive Patterns". In: *SDM*. 2012, pp. 236–247 (cit. on p. 47).

[165] Tiziano Squartini and Diego Garlaschelli. *Maximum-Entropy Networks: Pattern Detection, Network Reconstruction and Graph Combinatorics*. Springer, 2017 (cit. on p. 29).

[166] M. Sugiyama et al. "Significant Subgraph Mining with Multiple Testing Correction". In: *SDM*. 2015, pp. 37–45 (cit. on p. 98).

[167] Mahito Sugiyama and Karsten Borgwardt. "Finding Statistically Significant Interactions between Continuous Features". In: *IJCAI*. 2019 (cit. on p. 98).

[168] Mahito Sugiyama et al. "Significant Subgraph Mining with Multiple Testing Correction". In: *Proceedings of the 2015 SIAM International Conference on Data Mining, Vancouver, BC, Canada, April 30 - May 2, 2015*. 2015, pp. 37–45 (cit. on pp. 61, 69).

[169] Kaustubh Supekar et al. “Brain hyperconnectivity in children with autism and its links to social deficits”. In: *Cell Reports* 5.3 (2013), pp. 738–747 (cit. on p. 76).

[170] Tobias Sutter et al. “Generalized Maximum Entropy Estimation”. In: *J. Mach. Learn. Res.* 20 (2019), 138:1–138:29 (cit. on p. 134).

[171] R. E. Tarone. “A Modified Bonferroni Method for Discrete Data”. In: *Biometrics* 46.2 (1990), p. 515 (cit. on pp. 93, 99).

[172] Nikolaj Tatti. “Computational complexity of queries based on itemsets”. In: *Inf. Process. Lett.* 98.5 (2006), pp. 183–187 (cit. on pp. 116, 133).

[173] Nikolaj Tatti and Pauli Miettinen. “Boolean matrix factorization meets consecutive ones property”. In: *Proceedings of the 2019 SIAM International Conference on Data Mining, SDM 2019, Calgary, Alberta, Canada, May 2-4, 2019*. 2019, pp. 729–737 (cit. on p. 157).

[174] Nikolaj Tatti and Jilles Vreeken. “Comparing Apples and Oranges - Measuring Differences between Data Mining Results”. In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2011, Athens, Greece, September 5-9, 2011, Proceedings, Part III*. Vol. 6913. 2011, pp. 398–413 (cit. on p. 116).

[175] Aika Terada, Koji Tsuda, and Jun Sese. “Fast Westfall-Young permutation procedure for combinatorial regulation discovery”. In: *BIBM*. 2013, pp. 153–158 (cit. on pp. 86, 98, 99).

[176] Jinjin Tian and Aaditya Ramdas. “ADDIS: an adaptive discarding algorithm for online FDR control with conservative nulls”. In: *NeurIPS*. 2019, pp. 9383–9391 (cit. on p. 112).

[177] Jinjin Tian and Aaditya Ramdas. “Online control of the familywise error rate”. In: *Stat. Meth. Med. R.* 30.4 (2021), pp. 976–993 (cit. on p. 112).

[178] Martijn P van den Heuvel and Olaf Sporns. “A cross-disorder connectome landscape of brain dysconnectivity”. In: *Nature reviews neuroscience* 20.7 (2019), pp. 435–446 (cit. on p. 13).

[179] Fabio Vandin, Eli Upfal, and Benjamin J. Raphael. “Algorithms for Detecting Significantly Mutated Pathways in Cancer”. In: *J. Comput. Biol.* 18.3 (2011), pp. 507–522 (cit. on pp. 86, 98).

[180] Joshua T. Vogelstein et al. “Graph Classification Using Signal-Subgraphs: Applications in Statistical Connectomics”. In: *IEEE Trans. Pattern Anal. Mach. Intell.* 35.7 (2013), pp. 1539–1551 (cit. on pp. 69, 70).

[181] Jilles Vreeken, Matthijs v. Leeuwen, and Arno Siebes. “Krimp: mining itemsets that compress”. In: *Data Min. Knowl. Discov.* 23.1 (2011), pp. 169–214 (cit. on p. 98).

[182] Quang H Vuong. “Likelihood ratio tests for model selection and non-nested hypotheses”. In: *Econometrica: Journal of the Econometric Society* (1989), pp. 307–333 (cit. on p. 65).

[183] Changlin Wan et al. “Fast and Efficient Boolean Matrix Factorization by Geometric Segmentation”. In: *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, New York, NY, USA, February 7-12, 2020*. 2020, pp. 6086–6093 (cit. on p. 157).

[184] Lu Wang, Zhengwu Zhang, David Dunson, et al. “Common and individual structure of brain networks”. In: *AnnalsApplied-Statistics* 13.1 (2019), pp. 85–112 (cit. on p. 69).

[185] Lucy Lu Wang et al. “CORD-19: The Covid-19 Open Research Dataset”. In: *arXiv preprint arXiv:2004.10706* abs/2004.10706 (2020) (cit. on p. 143).

[186] Geoffrey I. Webb. “Layered Critical Values: A Powerful Direct-Adjustment Approach to Discovering Significant Patterns”. In: *Mach. Learn.* 71.2-3 (2008), pp. 307–323 (cit. on pp. 93, 99).

[187] Geoffrey I. Webb. “Self-sufficient itemsets: An approach to screening potentially interesting associations between items”. In: *ACM Trans. Knowl. Discov. Data* 4.1 (2010), 3:1–3:20 (cit. on pp. 47, 141).

[188] Geoffrey I. Webb and François Petitjean. “A Multiple Test Correction for Streams and Cascades of Statistical Hypothesis Tests”. In: *KDD*. 2016, pp. 1255–1264 (cit. on pp. 93, 99, 112).

[189] Geoffrey I. Webb and Jilles Vreeken. “Efficient Discovery of the Most Interesting Associations”. In: *ACM Trans. Knowl. Discov. Data* 8.3 (2013) (cit. on pp. 47, 141).

[190] Geoffrey I. Webb and Jilles Vreeken. “Efficient Discovery of the Most Interesting Associations”. In: *ACM Trans. Knowl. Discov. Data* 8.3 (2013), 15:1–15:31 (cit. on pp. 99, 106).

[191] Martin Weigt et al. “Identification of direct residue contacts in protein–protein interaction by message passing”. In: *Proceedings of the National Academy of Sciences* 106.1 (2009), pp. 67–72 (cit. on p. 29).

[192] World Bank. *World Integrated Trade Solution*. https://wits.worldbank.org/Default.aspx?lang=en. 2021 (cit. on p. 80).

[193] Hao Wu et al. “Generating Realistic Synthetic Population Datasets”. In: *ACM Trans. Knowl. Discov. Data* 12.4 (2018), 45:1–45:22 (cit. on pp. 116, 134).

[194] Hao Wu et al. “Uncovering the plot: detecting surprising coalitions of entities in multi-relational schemas”. In: *Data Mining and Knowledge Discovery* 28.5 (2014), pp. 1398–1428 (cit. on p. 29).

[195] Tianyi Wu, Shinya Sugawara, and Kenji Yamanishi. “Decomposed Normalized Maximum Likelihood Codelength Criterion for Selecting Hierarchical Latent Variable Models”. In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*. 2017, pp. 1165–1174 (cit. on pp. 46, 164).

[196] Wen Jun Xie, Mojgan Asadi, and Arieh Warshel. “Enhancing computational enzyme design by a maximum entropy strategy”. In: *Proceedings of the National Academy of Sciences* 119.7 (2022) (cit. on p. 29).

[197] Yujun Yan et al. “GroupINN: Grouping-based Interpretable Neural Network for Classification of Limited, Noisy Brain Data”. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*. 2019, pp. 772–782 (cit. on pp. 69, 70).

[198] Qingrun Zhang, Quan Long, and Jurg Ott. “AprioriGWAS, a New Pattern Mining Strategy for Detecting Genetic Variants Associated with Disease through Interaction Effects”. In: *PLoS* 10.6 (2014), p. 14 (cit. on pp. 86, 98).

[199] Zhongyuan Zhang et al. “Binary Matrix Factorization with Applications”. In: *Proceedings of the 7th IEEE International Conference on Data Mining (ICDM 2007), October 28-31, 2007, Omaha, Nebraska, USA*. 2007, pp. 391–400 (cit. on pp. 149, 157).

[200] Rui Zhao, Xudong Sun, and Volker Tresp. "Maximum Entropy-Regularized Multi-Goal Reinforcement Learning". In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*. Vol. 97. 2019, pp. 7553–7562 (cit. on p. 116).

[201] Hui Zou and Trevor Hastie. "Regularization and variable selection via the elastic net". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67.2 (2005), pp. 301–320 (cit. on pp. 147, 149).