Università
degli Studi
di Padova

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Master's Degree in Computer Engineering

Frequent Subgraph Mining via Sampling with Rigorous Guarantees

Candidate:

Paolo Pellizzoni

Supervisor:

Prof. **Fabio Vandin**

July 18, 2022

Department of Information Engineering
University of Padova
Padova, Italy

Academic Year 2021/2022

Abstract

Frequent subgraph mining is a fundamental task in the analysis of collections of graphs that aims at finding all the subgraphs that appear with more than a user-specified frequency in the dataset. While several exact approaches have been proposed to solve the task, it remains computationally challenging on large graph datasets due to the complexity of the subgraph isomorphism problem inherent in the task and the huge number of candidate patterns even for fairly small subgraphs.

In this thesis, we study two statistical learning measures of complexity, VC-dimension and Rademacher averages, for subgraphs, and derive efficiently computable bounds for both. We then show how such bounds can be applied to devise efficient sampling-based approaches for rigorously approximating the solutions of the frequent subgraph mining problem, providing sample sizes which are much tighter than what would be obtained by a straightforward application of Chernoff and union bounds. We also show that our bounds can be used for true frequent subgraph mining, which requires to identify subgraphs generated with probability above a given threshold using samples

from an unknown generative process.

Moreover, we carried out an extensive experimental evaluation of our methods on real datasets, which shows that our bounds lead to efficiently computable and high-quality approximations for both applications.

Contents

Chapter 1

Introduction

In many scientific domains, graphs are the key structure to represent complex data. For example, in chemistry and biology, structures as molecules can be effectively represented as graphs by representing the atoms as nodes and the bonds as edges, while proteins can be encoded by considering the amino-acids as nodes and peptide bonds as edges. As another example, in the analysis of social networks, graphs are used to grasp the relationships between users by means of likes, follows and friendships. Finding recurrent patterns in these types of graph-encoded data is of paramount importance to better understand the processes that generate such data, as they may highlight some unexpectedly frequent structures that are linked to the inner-workings of the domain.

This need is addressed by the frequent subgraph mining task, which is a fundamental data mining task that requires to identify small connected subgraphs appearing frequently in a collection of graphs. As stated before, it finds applications in a large number of domains, including social media marketing [FWWX15], graph classification and clustering [DKWK05, GK01],

recommendation systems for video games [ALA19], and computational biology [MMB+18].

The discovery of frequent subgraphs is a challenging task, due mostly to two reasons. First, to assess whether a subgraph appears in a graph requires to solve the subgraph isomorphism problem, which is, in general, NP-complete. Second, the number of candidate subgraphs is huge even for relatively small patterns. As a consequence, a number of exact approaches have been proposed [IWM00, YH02, NK05, KK01, BB02, WWP+04]. However, such exact approaches do not scale to large datasets.

Random sampling is a simple yet powerful technique to speed-up pattern mining, which has been applied to obtain *approximate* solutions for several patterns, such as itemsets [RU14], subgroups [RV20], and sequential patterns [STV20]. The major challenge in sampling approaches is to rigorously relate the results obtained from the analysis of the sample with the results that would be obtained analyzing the whole dataset, identifying sample sizes that provide rigorous guarantees on the quality of the approximation obtained analyzing the sample.

Recently, advanced tools from statistical learning theory, such as the Vapnik-Chervonenkis (VC) dimension [Vap98] and Rademacher averages [KP00], have been successfully used to obtain meaningful sample size required by random sampling for several pattern mining tasks [RU14, RU15, SSRZ20, STV20, PCVR20]. These tools improve over the use of standard approaches (e.g., using a Chernoff or Hoeffding bound for a single pattern followed by a union bound on all patterns), which often lead to sample sizes that are *larger* than the original dataset. To the best of our knowledge, tools from statistical learning theory have not been applied to mining frequent subgraphs from a collection of graphs.

1.1 Related work

In this Section we review the works related to this thesis. Subsection 1.1.1 provides an overview of the results of statistical learning theory related

to uniform deviation bounds. Subsection 1.1.2 reviews some works where approximation algorithms have been developed for patter mining problems, while Subsection 1.1.3 provides a summary on the state-of-the-art solvers for the frequent subgraph mining problem.

1.1.1 Statistical Learning Theory and Uniform Deviation Bounds

In the context of machine learning, a classical way to select the predictor (i.e. the trained machine learning model) is via empirical error minimization, that is to select the model that minimizes the loss function over the training set, in the hope that its loss over the unknown data generating distribution, which is the quantity that the user is really interested in, is not significantly worse. A common statistical learning tool to investigate whether such a hope is satisfied or not is *uniform convergence*, which informally is the property that, for each possible predictor in the chosen hypothesis class, for a high enough number of training samples, the empirical loss over the training dataset can be made arbitrarily close to the expected loss over the generating distribution [SSBD14, BBL05]. This problem can be seen, in general, as finding uniform (i.e. not depending on the specific hypothesis) deviation bounds of empirical averages of functions (in this case, losses) from their expectations, a formulation that is useful to apply this framework outside of machine learning.

If the hypothesis class is finite, the rate of uniform convergence can be bounded making use of standard tail bounds such as Hoeffding's inequality and an union bound over all hypotheses [SSBD14]. If the hypothesis class in infinite though, this analysis falls apart. In the seminal work of Vapnik and Chervonenkis [VC13, VC82], the authors defined the concept of VC dimension and its relation with uniform convergence, which was initially applied only to the convergence of probability distributions. Later, it was applied to the field of machine learning [BEHW89], showing that even when working with infinite hypothesis classes, provided that they are simple enough, one can obtain the uniform convergence property. Since then, the VC dimension has been widely

used to develop approximation algorithms in the fields of computational geometry [HS11], clustering [FL11, FSS20], database management [RAÇ+11] and pattern mining, which is covered in detail in the next section.

More advanced results on the rate of convergence in statistical learning are based on Rademacher averages [SSBD14, BBL05], which were introduced in the context of classification in [KP00]. Recently, Rademacher averages were used for developing approximate algorithms in the context of graph analytics [RU18] and pattern mining [RU15].

1.1.2 Approximate Pattern Mining

The problem of frequent pattern mining was introduced by Agrawal [AIS93] in the context of itemsets, and a wide variety of exact algorithms for solving the problem has risen over the years, the most well-known ones being Apriori [AS94] and FPGrowth [HPYM04].

Since the frequent itemset problem is computationally expensive, almost immediately some approximate algorithms were proposed to speed the computation up. Mannila et al. [MTV94] proposed an approximate mining algorithm based on sampling uniformly the transactions of the dataset, and used a combination of Chernoff bounds and union bounds to analyze the quality of the approximate solution in function of the number of samples. Since the quality of these theoretical bounds was quite unsatisfactory, other subsequent works [JL96, CB11] focused on developing heuristic progressive sampling algorithms, that work by enlarging a random sample until a simple to test termination condition is met. The main downside of these algorithms though is that the termination conditions on which they are based are heuristic, and thus do not offer any guarantee on the quality of the final solution.

The seminal work of Riondato et al. [RU14] introduced the use of tools from statistical learning theory in the context of frequent itemset mining, in particular by using the VC dimension to get a tighter analysis on the quality of the solution which avoided the use of union bound. They also provide polynomial-time algorithms to compute a tight upper bound on the VC dimension of the transaction dataset, as computing it exactly would be

extremely expensive. Indeed, they show that the VC dimenson of the dataset is upper bounded by the so-called *d-index*, that is the maximum number d such that there are d transactions in the dataset, each with at least d items, such that they form an anti-chain, i.e. that no such transaction is a subset of another one. While this quantity can be found in polynomial time, in modern datasets it is still too computationally expensive to compute, so the authors propose a linear time algorithm to find a looser upper bound, where the constraint on the transactions being an anti-chain is removed.

In a subsequent article [RU15], they developed a progressive sampling algorithm that uses another tool from machine learning theory, Rademacher averages, to provide guarantees when testing the termination condition. Indeed, using Rademacher averages they are able to bound the maximum deviation of the itemset frequencies in the sample from their true frequencies in the entire dataset. This approach allows to avoid processing the entire dataset, but rather only a small sample of it, and still obtain rigorous guarantees on the quality of the solution, which previous progressive sampling algorithms were not able to achieve. Since the Rademacher average computation is very computationally expensive, as in the case of frequent itemsets it involves computing the support for all the itemsets, they develop sharp bounds based on a refinement of the well-known Massart's lemma, which can be computed in time linear in the size of the sample.

In [RV14], the authors tackled the problem of finding frequent itemset in a slightly different setting. In particular, they consider the dataset as a sample from a unknown generating distribution π, and are interested in finding the itemsets that are generated frequently, that is with frequency at least θ, by π (i.e. the *true* frequent itemsets), rather than the ones that are frequent in the sample itself. Indeed, the set of frequent itemsets in the sample, due to the stochasticity of the generating process, contains many false positives, i.e. itemsets that are not really frequent in the generating distribution. Using an analysis based on the VC dimension of the dataset, they obtain rigorous bounds on the maximum deviation of the sample frequencies from the true ones, ad infer a refined threshold θ', as a function of the number of samples in the dataset, at which to mine the itemsets to avoid obtaining false positives.

Sampling has successfully used to obtain approximation algorithms for a wide variety of other pattern mining tasks. In a recent work [STV20], the authors employed a sampling approach to approximate frequent sequential pattern mining, which is a generalization of frequent itemset mining as sequential patterns are finite ordered sequences of itemsets, which have applications in domains such as e-commerce and biology. The analysis of the quality of the solutions follows the previously cited works, as it uses both the VC dimension and Rademacher averages to bound the maximum deviation of pattern frequencies in the subsampled dataset.

Yet another variation of pattern mining is interesting subgroups mining, that is to find all the subsets of a dataset for which the distribution of a specific target feature differs significantly from the distribution of such feature in the entire dataset. In [RV20], a random sampling algorithm is proposed to provide approximate solutions while speeding up the computation significantly. The analysis of the algorithm relies on yet another tool from statistical learning theory, the pseudodimension [Pol12], which is a generalization of the VC dimension to real-valued functions.

To the best of our knowledge, there is no work on rigorously approximating the frequent subgraph mining problem using sampling approaches. In the next subsection, we describe some of the exact algorithms for this problem.

1.1.3 Frequent Subgraph Mining

Driven by applications in fields such as computational chemistry and bioinformatics, the task of mining subgraphs from graph datasets has recently seen a lot of attention from the data mining community.

Overall, all frequent subgraph miners follow an approach similar to frequent itemset miners, that is they test the appearance of subgraphs in the dataset by exploring gradually the lattice formed by all possible patterns, pruning the branches of the search space that are not needed to be explored (e.g., thanks to the antimonotonicy property of frequency) along the way. The various miners differ in the order in which they explore the search space, in how they generate and represent the candidate patterns, and in how they

test the appearance of a pattern in the transactions of the dataset. We now describe briefly some of the most frequently used frequent subgraph miners up to date.

AGM [IWM00], which stands for Apriori Graph Mining, was one of the first subgraph miners and works similarly to Apriori, generating graph candidates adding one vertex at a time and for each one computing the frequency by scanning the dataset.

gSpan [YH02] explores the lattice in a dfs manner, using a canonical representation for graphs based on the dfs-traversal of the graph that helps in pruning the search space.

FFSM [HWP03] instead represents graphs by their adjaciency matrix, with some precautions to ensure that isomorphic graphs have the same representation. FFSM explores the lattice in a depth first search and stores the previous embeddings to avoid explicit subgraph isomorphism testing on the subsequent candidate patterns.

Gaston [NK05], which is reportedly the fastest subgraph miner up to date [WMFP05], exploits the fact that many frequent subgraphs are actually either paths or trees. Indeed, performing isomorphism checks on these graphs can be done efficiently in polynomial time, making much faster a large fraction of the mining process. Only at the end, when all acyclic frequent patterns have been found, the cyclic subgraphs are tested.

Among other frequent subgraph miners, we mention FSG [KK01], which improved on AGM, MoFa [BB02] which focuses mainly on molecular datasets, and ADI-Mine [WWP+04], which was the first miner to work in a disk-based setting, allowing to mine also datasets that could not fit in main memory.

Frequent subgraph mining is, in general, a harder task compared to frequent itemset mining, both because the number of possible patterns to be tested is larger and because each membership test involves a subgraph isomorphism check, which is notoriously an NP-hard problem. It is then very useful to develop approximation algorithms that, allowing for some slackness in the frequency threshold, are able to significantly reduce the computation time and the memory usage.

Another approach for dealing with the intrinsic complexity of the task

is to develop parallel and distributed algorithms. SUBDUE [CHGM01] is a shared-memory parallel frequent subgraph mining algorithm that uses a data partitioning approach to distribute the load across workers. It uses a heuristic beam search method that makes use of domain knowledge to prune the search space, which is explored in a bfs traversal. In the MapReduce framework, [HSS12] presented an algorithm that iteratively grows the candidate subgraph size in each round, until all frequent patterns have been found. This though results in a high number of rounds, and consequently in a high communication cost. Later, [LXG14] presented another MapReduce-based miner that works in 2 rounds by partitioning the transactions among the workers and computing locally frequent subgraphs in the first round and then refining the output in the second one.

The frequent subgraph mining problem has also a variant where we are given a single large graph and we are interested in finding the subgraphs that appear frequently within the large graph. Several exact and approximate algorithms have been developed also for this version of the problem [EASK14, GC02]. Recently, the VC dimension has been used to produce approximate solutions for this variant of the problem [PDFMR21], focusing on the Minimum Node Image (MNI) frequency measure for subgraphs. The techniques applied in [PDFMR21] though cannot be easily adapted to our scenario of a collection of graphs and its related notion of frequency for subgraphs.

1.2 Contributions

The contributions of this work are the following:

- We review in detail the known results in the literature on uniform deviation bounds, proposing a point of view that unifies the theory of range spaces and ε-approximations, which is well-known in the computational geometry community, an the one of Rademacher-average-based uniform convergence, which is more widespread in the machine learning community.
- We derive rigorous bounds on the VC-dimension of the class of subgraph

patterns that are efficiently computable for large datasets, improving the results that can obtained adapting the techniques previously proposes for other types of patterns. The computation of the bound requires only a single scan of the dataset and can be implemented in a streaming fashion, making its memory consumption negligible.

- We prove rigorous bounds on Rademacher averages for subgraph patterns, deriving different bounds specifically tailored for labelled graphs and for unlabelled graphs making use of a refined version of Massart's lemma.

- We show how our bounds on the VC-dimension can be used to design an effective algorithm to obtain approximate solutions with rigorous guarantees for the frequent subgraph mining problem, guaranteeing no false positives in the output (or no false negatives, depending on the user's choice). In a first step the algorithm derives the bound and samples the dataset, and in a second step it mines the sample, which can be carried out with any subgraph mining solver.

- Making use of our bound on Rademacher averages of subgraphs, we present a progressive sampling algorithm for the frequent subgraph mining problem, which builds incrementally a random sample of the dataset. This allows to avoid processing the entire dataset, making it possibly advantageous when its size is extremely large. As for the previous algorithm, in a first step it derives the bound and samples the dataset, and then it mines the sample with any subgraph mining solver.

- Using both the VC dimension and Rademacher averages, we develop two methods for mining true frequent subgraphs from an unknown generating distribution, a variant which requires to identify the subgraphs appearing with probability greater than a given threshold in a generative process having as input a set of samples from the process.

- For all of the algorithms above, we performed an experimental evaluation on real-life graph datasets to test the usability of the methods in a

practical environment. From the experiments it emerges that, when we allow for a mere 1% of slackness in the value of the frequency threshold, we gain up to a 40x improvement in terms of execution time.

1.3 Roadmap

The thesis is organized as follows. Chapter 2 provides all the preliminaries necessary to tackle the problem, and in particular reviews in detail the results on uniform deviation bounds, Chapter 3 presents an efficiently computable bound on the VC dimension of subgraphs, while Chapter 4 details a novel bound on Rademacher averages of frequent subgraphs. Chapter 5 presents two applications of our novel bounds, for the tasks of frequent subgraph mining (Section 5.1) and of true frequent subgraph mining (Section 5.2). In Chapter 6 some experiments on the proposed methods are presented, along with a discussion of the results.

Chapter 2

Preliminaries

In this chapter we introduce all the notions needed to tackle the problem at hand. In particular, Section 2.1 defines formally the problem of frequent subgraph mining. Section 2.2 describes the notions of range spaces, of VC-dimension and how it is applied to sampling to obtain approximation algorithms. Finally, Section 2.3 deals with Rademacher averages, another tool from statistical learning theory that can be employed to obtain approximation algorithms in data mining problems.

2.1 Frequent Subgraph Mining

We define a dataset $\mathcal{D}$ as a collection of unweighted, undirected and labelled graphs $G = (V, E, L_V, L_E)$, where L_V and L_E are functions that map nodes and edges respectively to fixed labels. We refer to such graphs as *transactions* of the dataset. Two graphs G and G' are isomorphic if there exists a bijection μ from the nodes of the first one to the ones of the second one that preserves

the node labels and s.t. $(u,v) \in E \iff (\mu(u),\mu(v)) \in E'$ and $L_E(u,v) = L_E(\mu(u),\mu(v))$. We say that graph H is isomorphic to an induced subgraph of G if there is an induced subgraph of G isomorphic to H. In $\mathcal{D}$ there can be isomorhpic graphs.

Let $\mathcal{P}$ be the *pattern* set, that is a set of connected graphs whose frequency in the dataset is of interest. The set $\mathcal{P}$ can be the set of all connected graphs, the set of all connected graphs with up to k nodes, or a specific language of patterns of interest, although the mining algorithms have to be suitably modified for this last case. We say that G contains P, denoted with $P \subseteq G$, if P is isomorphic to an induced subgraph of G.

Given a pattern $P \in \mathcal{P}$, the support set $T_{\mathcal{D}}(P)$ of P is the set of transactions in $\mathcal{D}$ that contain the pattern P. The frequency of P is the fraction of transactions that contain P, $f_{\mathcal{D}}(P) = |T_{\mathcal{D}}(P)|/|\mathcal{D}|$.

Definition 1. *Given a frequency threshold θ, the frequent subgraph mining task is to find all patterns with frequency above the threshold, along with their frequencies, that is*

$$FG(\mathcal{D},\mathcal{P},\theta) = \{(P, f_{\mathcal{D}}(P)) : \ P \in \mathcal{P} \text{ and } f_{\mathcal{D}}(P) \geq \theta\}.$$

Since often the threshold θ is chosen arbitrarily, one often can sacrifice solving the problem exactly and settle for an approximate solution. A commonly used [CPS09] notion of approximation for frequent pattern mining, which guarantees no false negatives, is the following.

Definition 2. *Given a parameter $\varepsilon > 0$, an absolute ε-close solution to $FG(\mathcal{D},\mathcal{P},\theta)$ is a set $C = \{(P, \hat{f}(P)) : \ P \in \mathcal{P} \text{ and } \hat{f}(P) \in [0,1]\}$ such that*

1. *$FG(\mathcal{D},\mathcal{P},\theta) \subseteq C$*
2. *C contains no pattern P with frequency $f_{\mathcal{D}}(P) < \theta - \varepsilon$*

Actually, since in general it is useful to know the (approximate) frequency of the frequent patterns, we focus on a slightly more restrictive definition [RU14] of approximate solution for the frequent subgraph mining problem.

Definition 3. *Given a parameter $\varepsilon > 0$, an absolute ε-close approximation to $FG(\mathcal{D}, \mathcal{P}, \theta)$ is a set $C = \{(P, \hat{f}(P)) : \ P \in \mathcal{P}$ and $\hat{f}(P) \in [0, 1]\}$ such that*

1. *$FG(\mathcal{D}, \mathcal{P}, \theta) \subseteq C$*
2. *C contains no pattern P with frequency $f_{\mathcal{D}}(P) < \theta - \varepsilon$*
3. *for each $(P, \hat{f}(P))$ it holds $|\hat{f}(P) - f_{\mathcal{D}}(P)| \leq \varepsilon$*

Note that the definitions above produce approximate solutions with no false negatives, but we can easily define a notion of approximation with no false positives. For the sake of clarity and brevity, in the subsequent chapters we will only focus on the definition with no false negatives.

Another problem we consider is the mining of *true frequent subgraphs*, which adapts the problem of true frequent itemsets mining [RV14] to subgraphs. In this scenario, the dataset $\mathcal{D}$ consists of a number of graphs obtained from an unknown generative process, described by an (unknown) generating distribution π, with $p_\pi(G)$ being the probability that graph G is the graph generated by π. The probability that a subgraph P is in a sample from the distribution π is then $p_\pi(P) = \sum_{G:P \subseteq G} p_\pi(G)$.

Definition 4. *Given a frequency threshold θ, the true frequent subgraph mining task is to find all subgraphs with probability at least θ of appearing in the sample, along with such probabilities, that is*

$$FG(\pi, \mathcal{P}, \theta) = \{(P, p_\pi(P)) : \ P \in \mathcal{P} \text{ and } p_\pi(P) \geq \theta\}.$$

This problem is highly relevant when one is analyzing a datasets obtained as a sample from an underlying process and needs guarantees on the underlying process (instead that on the dataset). Note that the true frequent subgraph mining problem cannot be solved exactly without full knowledge of π. For true frequent subgraph mining, the definition of ε-close approximation is a simple adaptation of Def. 3, obtained replacing $FG(\mathcal{D}, \mathcal{P}, \theta)$ with $FG(\pi, \mathcal{P}, \theta)$ and $f_{\mathcal{D}}(P)$ with $p_\pi(G)$.

2.2 Range Spaces and ε-approximations

The VC dimension is a measure of the complexity of a family of indicator functions over a space of points which has been widely used in the context of machine learning. In this section we describe how to exploit this quantity to obtain approximate solutions to hard computational problems by using random sampling.

Definition 5. *We define a range space as a pair (X, R) where X is a set and R is a family of subsets of X. The projection of R on $A \subseteq X$ is $P_R(A) = \{r \cap A : r \in R\}$. If $P_R(A) = 2^A$ we say that A is shattered by R.*

The definition of range spaces is very general and allows to be suitably adapted to a wide variety of applications. For example, in the context of computational geometry, the set X is usually a subset of $\mathbb{R}^d$ and R is a family of simple shapes such as halfspaces, axis-aligned boxes or balls. In the context of machine learning, the set X is often yet again $\mathbb{R}^d$, the feature space of the samples, and the ranges are the functions of the hypothesis class $\mathcal{H}$ of interest, which can be as simple as the family of halfplanes for hard SVMs, but for models such as neural networks it can be very complex.

Range spaces can be endowed with a notion of complexity, the VC dimension, which captures how rich the family of ranges R is.

Definition 6. *Let (X, R) be a range space. The VC-dimension $VC(X, R)$ of (X, R) is the largest cardinality of a subset of X which is shattered by R. If R can shatter arbitrarily large subsets of X, then we define $VC(X, R) = +\infty$.*

If one has access only to a subset of the set X, one can resort to a slightly weaker notion of complexity, the empirical VC dimension.

Definition 7. *Let (X, R) be a range space and let $Y \subset X$. Then the empirical VC dimension of (X, R) on Y, denoted as $EVC((X, R), Y)$, is the VC dimension of the range space (Y, R'), with $R' = \{Y \cap r : r \in R\}$.*

A type of query on range spaces that is very relevant for the problem of frequent pattern mining is *range counting*, that is to return, for a range

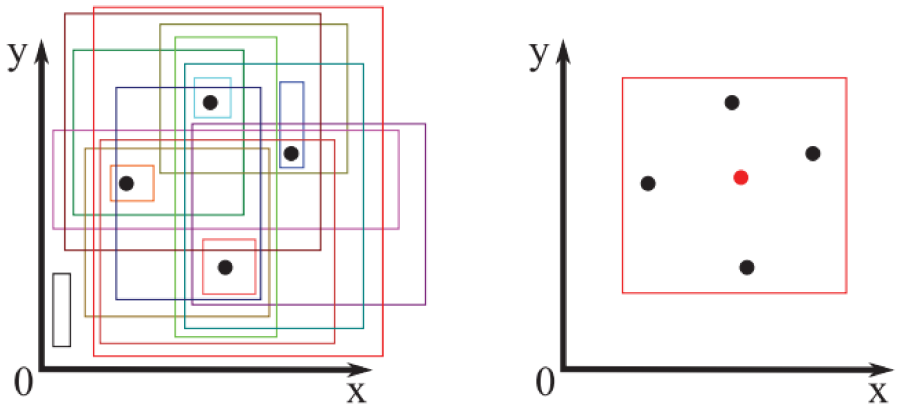

Figure 2.1: Example [RU14] of a range space, with $X = \mathbb{R}^2$ and R the set of axis-aligned rectangles. As shown on the left, it is possible to shatter a set of four points with R, so $VC(X, R) \geq 4$. On the right, instead, we can see a hint to the fact that, for any choice of five points in the plane, there will always be one set of four points that cannot be covered by a range without including the fifth one.

$r \in R$ and a set $A \subseteq X$, the quantity $\frac{|A \cap r|}{|A|}$. For example, if r is the range of transactions that contain a given pattern, the range counting query is actually to compute the frequency of the pattern in the set A of transactions. Another application of this type of query is in computational geometry. For example, in the classical halfspace range counting problem [HS11], one is asked, for a set of points A in $\mathbb{R}^d$ and an halfspace r, which can be seen as a range, to count the fraction of points lying within r.

One application of VC-dimension is to bound the number of samples needed to approximate range counting. A common notion of approximation is the following [HS11].

Definition 8. *Let (X, R) be a range space and let A be a finite subset of X. For $0 < \varepsilon < 1$, a subset $B \subset A$ is an ε-approximation for A if for all $r \in R$ we have*

$$\left| \frac{|A \cap r|}{|A|} - \frac{|B \cap r|}{|B|} \right| \leq \varepsilon.$$

A simple yet effective idea to obtain absolute ε-close approximations for the range counting problem is to use random sampling on the input dataset,

although it is not clear a priori how the number of samples should be chosen. In the case that R is finite, we have the following theorem.

Theorem 1. *Let (X, R) be a range space with finite R. If $A \subset X$ is a finite subset and $0 < \varepsilon, \delta < 1$, then a bag B of elements of A, taken uniformly at random and with replacements, of cardinality*

$$m = \min\left\{|A|, \frac{1}{2\varepsilon^2}\ln\left(\frac{2|R|}{\delta}\right)\right\}$$

is an ε-approximation for A with probability $1 - \delta$. If $m = |A|$ we assume $B = A$.

Proof. Hoeffding's inequality states that, if $X_1, \ldots, X_n$ are i.i.d. random variables that take values in $[a, b]$ almost surely, then

$$\mathbb{P}\left[\left|\frac{1}{m}\sum_{i=1}^{m} X_i - \mathbb{E}[X_1]\right| > \varepsilon\right] \leq 2\exp\left(-\frac{2m\varepsilon^2}{(b-a)^2}\right).$$

Let $B = (b_1, \ldots, b_n)$. For a fixed range $r \in R$, let $X_i = 1$ if $b_i \in r$ and 0 otherwise. Then we have that $X_i \in [0, 1]$, $\mathbb{E}[X_i] = \frac{|A \cap r|}{|A|}$ and $\frac{1}{m}\sum_{i=1}^{m} X_i = \frac{|B \cap r|}{|B|}$. Then we have

$$\mathbb{P}\left[\left|\frac{1}{m}\sum_{i=1}^{m} X_i - \mathbb{E}[X_1]\right| > \varepsilon \,\middle|\, r\right] \leq 2\exp\left(-2m\varepsilon^2\right).$$

Taking union bound over all $r \in R$ we have

$$\mathbb{P}\left[\left|\frac{1}{m}\sum_{i=1}^{m} X_i - \mathbb{E}[X_1]\right| > \varepsilon\right] \leq 2|R|\exp\left(-2m\varepsilon^2\right).$$

Therefore, if we take $m \geq \frac{1}{2\varepsilon^2}\ln\left(\frac{2|R|}{\delta}\right)$ samples from A, we have, with probability at least $1 - \delta$, that $\left|\frac{|A \cap r|}{|A|} - \frac{|B \cap r|}{|B|}\right| \leq \varepsilon$, that is B is an ε-approximation for A.

□

In our setting, the theorem above has limited utility, since in most cases the set R is huge or even infinite. The next theorem [HS11] links the notion

of VC-dimension and ε-approximations, providing much tighter bounds.

Theorem 2. *There is an absolute positive constant c such that if (X, R) is a range space of VC dimension at most v, $A \subset X$ is a finite subset and $0 < \varepsilon, \delta < 1$, then a bag B of elements of A, taken uniformly at random and with replacements, of cardinality*

$$m = \min\left\{|A|, \frac{c}{\varepsilon^2}\left(v + \log(1/\delta)\right)\right\}$$

is an ε-approximation for A with probability $1 - \delta$. If $m = |A|$ we assume $B = A$.

Although no theoretical result upper bounding c is known as of now, thanks to some experimental evidence presented in [LP09], the constant c is usually considered to be close to 0.5.

Actually, we can define a more general notion of ε-approximation that works with distributions on X.

Definition 9. *Let (X, R) be a range space and let π a distribution on X. For $0 < \varepsilon < 1$, a bag B of elements of X is an ε-approximation for (X, π) if, for all $r \in R$ we have*

$$\left| p_\pi(r) - \frac{1}{|B|} \sum_{x \in B} \mathbf{1}_r(x) \right| \leq \varepsilon$$

where $p_\pi(r) = \sum_{x \in r} p_\pi(x)$ and $\mathbf{1}_r$ is the indicator function for the range r.

The results of Theorem 2 hold even for this general definition of ε-approximations, albeit with the caveat that sampling has to be performed according to π rather than uniformly at random.

2.3 Rademacher Averages

Rademacher averages are a tool from statistical learning theory widely used to characterize the rate of uniform convergence in PAC learning. We now

expose some of the central bounds on uniform deviations of empirical means from their expected values which use this tool.

Let X_1, ..., X_n be i.i.d. random variables taking values in a set $\mathcal{X}$ and let $\mathcal{F}$ be a family of bounded functions $f : \mathcal{X} \longrightarrow [0, 1]$. Let $\hat{f}(X) = \frac{1}{n}\sum_{i=1}^{n} f(X_i)$. We wish to bound the maximum deviation

$$\sup_{f\in\mathcal{F}} \left|\mathbb{E}[f(X_1)] - \hat{f}(X)\right|.$$

We'll make use of the following concentration inequality, which is also known as McDiarmid's inequality [McD89].

Theorem 3 (Bounded differences inequality)**.** *Let $g : \mathcal{X}^n \longrightarrow \mathbb{R}$ be a function such that, for some nonnegative constants $c_1, \ldots, c_n$,*

$$\sup_{x_1,\ldots,x_n,x_i'\in\mathcal{X}} |g(x_1, \ldots, x_n) - g(x_1, \ldots, x_{i-1}, x_i', x_{i+1}, \ldots, x_n)| \leq c_i,\ 1 \leq i \leq n.$$

Let X_1, ..., X_n be independent random variables. Then the random variable $Z = g(X_1, \ldots, X_n)$ satisfies

$$\mathbb{P}[|Z - \mathbb{E}[Z]| > t] \leq 2\exp\left(-\frac{-2t^2}{\sum_{i=1}^{n} c_i^2}\right).$$

Equivalently, we have that with probability at least $1 - \delta$,

$$|Z - \mathbb{E}[Z]| < \sqrt{\frac{\sum_{i=1}^{n} c_i^2}{2}\ln\left(\frac{2}{\delta}\right)}.$$

As a consequence of the bounded difference inequality, if we let $Z = \sup_{f\in\mathcal{F}} \left|\mathbb{E}[f(X_1)] - \hat{f}(X)\right|$, which satisfies the inequality with $c_i = 1/n$, we have that, for any $\delta \in]0, 1[$, with probability at least $1 - \delta$,

$$\sup_{f\in\mathcal{F}} \left|\mathbb{E}[f(X_1)] - \hat{f}(X)\right| \leq \mathbb{E}\left[\sup_{f\in\mathcal{F}} \left|\mathbb{E}[f(X_1)] - \hat{f}(X)\right|\right] + \sqrt{\frac{\ln\frac{1}{\delta}}{2n}}.$$

We then proceed to bound the right hand side of the inequality. We first define Rademacher averages.

Definition 10. *Let $A \subset \mathbb{R}^n$ be a set of bounded vectors $a = (a_1, \ldots, a_n)^T$, $a_i \in [0, 1]$. Then let σ_i be a Rademacher random variable, i.e. taking value 1 or -1 with probability $1/2$, for each $1 \leq i \leq n$. Let also the σ_i's be independent. The (sample) conditional Rademacher average, also known as the empirical Rademacher average, is the quantity*

$$\mathcal{R}(A) = \mathbb{E}_\sigma \left[\sup_{a \in A} \frac{1}{n} \sum_{i=1}^{n} \sigma_i a_i \right]$$

where $\mathbb{E}_\sigma$ denotes the expectation taken only with respect to the σ_i's, conditionally on the sample A.

In particular, if we call $\mathcal{F}(X) = \{(f(X_1), \ldots, f(X_n))^T : \ f \in \mathcal{F}\}$, then we have the following lemma [BBL05], which is based on a symmetrization argument.

Lemma 1.

$$\mathbb{E}\left[\sup_{f \in \mathcal{F}} \left| \mathbb{E}[f(X_1)] - \hat{f}(X) \right| \right] \leq 2\mathbb{E}_X \left[\mathcal{R}(\mathcal{F}(X)) \right].$$

In turn, noticing that $\mathcal{R}(\mathcal{F}(X))$ satisfies the bounded differences inequality, this yields the following result[1], that removes the need to compute an expectation over X.

Theorem 4. *With probability at least $1 - \delta$,*

$$\sup_{f \in \mathcal{F}} \left| \mathbb{E}[f(X_1)] - \hat{f}(X) \right| \leq 2\mathcal{R}(\mathcal{F}(X)) + 3\sqrt{\frac{\ln \frac{2}{\delta}}{2n}}.$$

Proof. We have that the random variable $\mathcal{R}(\mathcal{F}(X)) = g(X_1, \ldots, X_n)$ satisfies the bounded differences inequality with $c_i = 1/n$. Then we have, with

[1] In [BBL05], the theorem has a 1 rather than a 3 as the constant multiplying the square root. The paper is missing the details of the proof and we were not able to replicate the result.

probability $1 - 2\delta$,

$$\sup_{f \in \mathcal{F}} \left| \mathbb{E}[f(X_1)] - \hat{f}(X) \right| \leq \mathbb{E}\left[\sup_{f \in \mathcal{F}} \left(\mathbb{E}[f(X_1)] - \hat{f}(X) \right) \right] + \sqrt{\frac{\ln \frac{1}{\delta}}{2n}} \leq$$
$$\leq 2\mathbb{E}\left[\mathcal{R}(\mathcal{F}(x)) \right] + \sqrt{\frac{\ln \frac{1}{\delta}}{2n}} \leq$$
$$\leq 2\mathcal{R}(\mathcal{F}(x)) + 3\sqrt{\frac{\ln \frac{1}{\delta}}{2n}}.$$

□

The above theorem gives a data-dependent bound on the maximum deviation of the empirical average of f from the true expected value, over all possible $f \in \mathcal{F}$.

The following theorem, which is the main result of this Section, then links the notion of ε-approximations with Rademacher-averages-based bounds on uniform deviations of empirical averages from their expectations.

Theorem 5. *Consider a range space $(\mathcal{X}, R)$ and let π be a distribution over $\mathcal{X}$. Consider then as the function set $\mathcal{F}$ the set $\{f_r : r \in R\}$ of indicator functions for the ranges of R. Let $X_1, \ldots, X_n$ be i.i.d. random variables with distribution π and let $B = (x_1, \ldots, x_n)$ be the bag of realizations of the random variables. Let*

$$\varepsilon = 2\mathcal{R}(\mathcal{F}(X)) + 3\sqrt{\frac{\ln \frac{2}{\delta}}{2n}}.$$

Then, with probability at least $1 - \delta$, B is an ε-approximation for X w.r.t. π.

Proof. Then we have that $\mathbb{E}[f_r(X_1)] = p_\pi(r)$ and $\hat{f}_r(X) = \frac{1}{|B|} \sum_{x \in B} \mathbf{1}_r(x)$. Then, if we are able to bound the maximum deviation of the functions in $\mathcal{F}$ as $\sup_{f \in \mathcal{F}} \left| \mathbb{E}[f_r(X_1)] - \hat{f}_r(X) \right| \leq \varepsilon$, we can say that B is an ε-approximation for X w.r.t. π, by Definition 9. Indeed, thanks to Theorem 4, we have that with probability $1 - \delta$, choosing $\varepsilon = 2\mathcal{R}(\mathcal{F}(X)) + 3\sqrt{\frac{\ln \frac{2}{\delta}}{2n}}$ yields the above property. □

2.3.1 Bounding Rademacher Averages

Note that computing the quantity $\mathcal{R}(\mathcal{F}(X))$ is not a straightforward task, as it involves both computing a supremum over $\mathcal{F}$, which can be a very hard task to solve, and computing an expectation over the σ_i's, which usually requires some sort of Monte Carlo integration, which is in itself quite demanding. We then resort to developing upper bounds to it.

In the subsequent analysis, we use the well-known Jensen inequality. The probabilistic formulation of the inequality is as follows.

Lemma 2 (Jensen inequality)**.** *Let $\psi : \mathbb{R} \longrightarrow \mathbb{R}$ be a convex function and let X be a random variable taking values in $\mathbb{R}$. Then we have*

$$\psi\left(\mathbb{E}[X]\right) \leq \mathbb{E}\left(\psi(X)\right).$$

We then now present a powerful tool to bound Rademacher averages, namely Massart lemma [BBL05].

Lemma 3. *Let $A \subset \mathbb{R}^n$ be a finite set. Let $w : \mathbb{R}^+ \to \mathbb{R}^+$ be the function*

$$w(s) = \frac{1}{s} \ln\left(\sum_{a \in A} \exp\left(s^2 \left\|a\right\|^2 / (2n^2)\right)\right).$$

Then $\mathcal{R}(A) \leq \min_{s \in \mathbb{R}^+} w(s)$.

Proof. Hoeffding inequality states that if X is a zero-mean r.v. taking values in $[a, b]$, then for each $s > 0$, $\mathbb{E}\left[\exp\left(sX\right)\right] \leq \exp(s^2(b-a)^2/8)$. Then we have

$$\mathbb{E}\left[\exp\left(s\frac{1}{n}\sum_{i=1}^{n}\sigma_i a_i\right)\right] = \prod_{i=1}^{n}\mathbb{E}\left[\exp\left(s\frac{1}{n}\sigma_i a_i\right)\right] \leq \prod_{i=1}^{n}\exp\left(\frac{s^2 a_i^2}{2n^2}\right) =$$
$$= \exp\left(\frac{s^2\left\|a\right\|^2}{2n^2}\right).$$

In particular, we can use the inequality above to obtain

$$\exp(s\mathcal{R}(A)) = \exp\left(s\mathbb{E}\left[\max_{a\in A}\frac{1}{n}\sum_{i=1}^{n}\sigma_i a_i\right]\right) \leq \mathbb{E}\left[\exp\left(s\max_{a\in A}\frac{1}{n}\sum_{i=1}^{n}\sigma_i a_i\right)\right] \leq$$
$$\leq \sum_{a\in A}\mathbb{E}\left[\exp\left(s\frac{1}{n}\sum_{i=1}^{n}\sigma_i a_i\right)\right] \leq \sum_{a\in A}\exp\left(\frac{s^2\left\|a\right\|^2}{2n^2}\right).$$

The first inequality is obtained applying Jensen inequality, since e^x is convex. Taking the logarithm and dividing by s yields the result. □

The lemma is mostly known in the following weaker form, which can be easily obtained from the previous formulation.

Theorem 6 (Massart's Lemma)**.**

$$\mathcal{R}(A) \leq \max_{a\in A}\|a\|\frac{\sqrt{2\ln|A|}}{n}$$

Proof. We have that

$$w(s) = \frac{1}{s}\ln\left(\sum_{a\in A}\exp\left(s^2\left\|a\right\|^2/(2n^2)\right)\right) \leq$$
$$\leq \frac{1}{s}\ln\left(|A|\exp\left(\max_{a\in A}\left\|a\right\|^2\cdot s^2/(2n^2)\right)\right) =$$
$$= \frac{\ln|A|}{s} + s\cdot\frac{\max_{a\in A}\|a\|}{2n^2}.$$

Taking the minimum of the right hand side of the inequality over $s\in\mathbb{R}^+$ yields $w(s)\leq\max_{a\in A}\|a\|\frac{\sqrt{2\ln|A|}}{n}$, with the minimum taking place at $s=\frac{n\sqrt{2\ln|A|}}{\max_{a\in A}\|a\|}$. Then, using Lemma 3 we have the claim. □

2.3.2 Linking Rademacher Averages and VC Dimension

Consider a range space $(\mathcal{X}, R)$, a distribution π over $\mathcal{X}$, and as $\mathcal{F}$ the set $\{f_r : r\in R\}$ of indicator functions for the ranges of R. Let $X_1,\ldots,X_n$ be i.i.d. random variables with distribution π and let $B=(x_1,\ldots,x_n)$ be the realizations of the random variables.

By Theorem 6, recalling that $\max_{f\in\mathcal{F}} \|f(x)\| \leq \sqrt{n}$, we have that $\mathcal{R}(\mathcal{F}(X)) \leq \sqrt{\frac{2\ln|\mathcal{F}(B)|}{n}}$.

A trivial bound to $|\mathcal{F}(X)|$ is 2^n. We show that if the VC dimension of the range space is low, a better bound can be obtained. Let $d = EVC((X,R),B)$ be the empirical VC dimension of the range space $(\mathcal{X},R)$ on the bag $B = (x_1,\ldots,x_n)$ of realizations of the random variables $X_1,\ldots,X_n$.

Then, using Sauer's Lemma [BBL05], which links the size of a set to its VC dimension, we have

$$|\mathcal{F}(X)| \leq \sum_{i=0}^{d} \binom{n}{i} \leq (n+1)^d$$

which yields $\ln|\mathcal{F}(X)| \leq d\ln(n+1)$. Combining this result with Theorem 4, we have the following.

Lemma 4. *With probability at least* $1-\delta$*,*

$$\sup_{f\in\mathcal{F}} \left|\mathbb{E}[f(X_1)] - \hat{f}(X)\right| \leq 2\sqrt{\frac{2d\ln(n+1)}{n}} + 3\sqrt{\frac{\ln\frac{1}{\delta}}{2n}}.$$

This result can be then used to obtain, given a sample $B = (x_1,\ldots,x_n)$, the value ε for which B is an ε-approximation for $\mathcal{X}$ w.r.t. π. Note that this bound is weaker with respect to the one presented in Theorem 2, but it only uses sample-dependent quantities, so it can be applied even if one has no access to the VC dimension of $(\mathcal{X},R)$.

Chapter 3

VC Dimension of Subgraphs

As described in the previous chapters, the VC dimension is a measure of the complexity of a family of indicator functions over a space of points which has been widely used in the context of machine learning. In this section we exploit this quantity to circumvent the use of union bound and obtain tighter bounds on the number of samples necessary to obtain an ε-close approximation for frequent subgraph mining.

We now define a range space for the problem of frequent subgraph mining and use it to determine sample sizes sufficient to obtain approximate solutions within a desired slackness.

Definition 11. *Let $\mathcal{D}$ be a dataset of transactions and $\mathcal{P}$ a set of patterns. We define $(\mathcal{D}, R_{\mathcal{P}})$ as the range space such that:*

1. *$\mathcal{D}$ is the set of transactions*
2. *$R_{\mathcal{P}} = \{r_P = T_{\mathcal{D}}(P) : \ P \in \mathcal{P}\}$ is a family of sets of transactions (i.e., graphs) s.t. for each pattern $P \in \mathcal{P}, T_{\mathcal{D}}(P)$ is the set of transactions containing P.*

From the above definition we can clearly see that for a subset $\mathcal{D}'$ of $\mathcal{D}$, the quantity $\frac{|\mathcal{D}' \cap r_P|}{|\mathcal{D}'|} = \frac{|\mathcal{D}' \cap T_{\mathcal{D}}(P)|}{|\mathcal{D}'|}$ is $f_{\mathcal{D}'}(P)$, the frequency of P in $\mathcal{D}'$.

3.1 Bounding the VC Dimension

Computing the VC dimension of a range space exactly is a very expensive operation, as in general it takes $O\left(|R||X|^{\log |R|}\right)$ time [LMR91]. We have then to efficiently compute an upper bound to it in order to exploit Theorem 11.

We now define a characteristic of the dataset called the d-index [RU14], and show that it is an upper bound to the VC-dimension of the range space associated to the dataset.

Definition 12. *Let $\mathcal{D}$ be a dataset. The d-index of $\mathcal{D}$ is the maximum integer d such that $\mathcal{D}$ contains at least d different transactions (i.e., graphs) $G_i = (V_{G_i}, E_{G_i})$ such that $|V_{G_i}| \geq d$ and such that no one of them is isomorphic to an induced subgraph of another, that is they form an anti-chain.*

The following lemma is a simple adaptation of [RU14, Theorem 4.2] to a dataset of graphs.

Lemma 5. *Let $\mathcal{D}$ be a dataset of transactions and let $(\mathcal{D}, R_{\mathcal{P}})$ be the associated range space. Then $VC(\mathcal{D}, R_{\mathcal{P}}) \geq v$ if and only if there exists a set $\mathcal{A} \subseteq \mathcal{D}$ of v transactions such that for each subset $\mathcal{B} \subseteq \mathcal{A}$, there exists a pattern $P_{\mathcal{B}}$ such that the support set of $P_{\mathcal{B}}$ in $\mathcal{A}$ is exactly $\mathcal{B}$, i.e. $T_{\mathcal{A}}(P_{\mathcal{B}}) = \mathcal{B}$.*

Proof. If: We have that $T_{\mathcal{D}}(P_{\mathcal{B}}) \cap \mathcal{A} = \mathcal{B}$, for each of the $2^{|\mathcal{A}|}$ subsets $\mathcal{B}$ of $\mathcal{A}$. Hence $\mathcal{A}$ is shattered by $R_{\mathcal{P}}$ and $VC(\mathcal{D}, R_{\mathcal{P}}) \geq v$.

Only if: Let $VC(\mathcal{D}, R_{\mathcal{P}}) \geq v$. Then, there is a set $\mathcal{A} \subseteq \mathcal{D}$ of v transactions such that $P_{R_{\mathcal{P}}}(\mathcal{A}) = 2^{\mathcal{A}}$. Hence, for each subset $\mathcal{B}$ of $\mathcal{A}$, there exists a pattern $P_{\mathcal{B}}$ such that $T_{\mathcal{D}}(P_{\mathcal{B}}) \cap \mathcal{A} = \mathcal{B}$. □

Theorem 7. *Let $\mathcal{D}$ be a dataset with d-index d and $\mathcal{P}$ the pattern set. Then the range space $(\mathcal{D}, R_{\mathcal{P}})$ has VC-dimension at most d.*

Proof. Let $l > d$ and assume that $(\mathcal{D}, R_{\mathcal{P}})$ has VC-dimension l. Then there is a set of $\mathcal{A}$ of l transactions of $\mathcal{D}$ that is shattered by $R_{\mathcal{P}}$, that is for each

subset $\mathcal{B} \subseteq \mathcal{A}$, there exists a pattern $P_\mathcal{B}$ such that $T_\mathcal{A}(P_\mathcal{B}) = \mathcal{B}$. Then for any two transactions $G_1, G_2 \in \mathcal{A}$, neither one is isomorphic to an induced subgraph of the other. In fact, if $G_1 \subseteq G_2$, then in all ranges where G_1 appears, i.e. in all the sets $T(P)$ of transactions such that $P \subseteq G_1$, also G_2 appears. Indeed, if $P \subseteq G_1$ and $G_1 \subseteq G_2$, then also $P \subseteq G_2$, and hence $G_2 \in T(P)$. Then it would not be possible to shatter $\mathcal{A}$, as there would be no pattern P_{G_1} such that $T_\mathcal{A}(P_{G_1}) = \{G_1\}$. Then $\mathcal{A}$ must be an anti-chain. Therefore, $\mathcal{A}$ must contain a graph G such that $|V_G| \leq d$, or we would have d-index $> d$.

This graph G is a member of 2^{l-1} subsets of $\mathcal{A}$, which we call $\mathcal{A}_i$'s, labelled in any order. Since $\mathcal{A}$ is shattered by $R_\mathcal{P}$, we have that for each set $\mathcal{A}_i$, there exists a pattern P_i such that $T(P_i) \cap \mathcal{A} = \mathcal{A}_i$. Note that since the $\mathcal{A}_i$'s are all different, then also the P_i's must be all different. Since $G \in \mathcal{A}_i$ by construction, we must have that $G \in T(P_i)\ \forall 1 \leq i \leq 2^{l-1}$. Then all patterns P_i appear as an induced subgraph of G. But, since $|V_G| \leq d < l$, G can contain at most $2^d - 1$ non-empty patterns (recall that an induced subgraph is defined only by the subset of nodes of the original graph), while there are 2^{l-1} different patterns P_i. This is a contraddiction and hence $\mathcal{A}$ cannot be shattered by $R_\mathcal{P}$, so $VC(S) \leq d$.

□

The d-index can be computed exactly, with a number of subgraph isomorphism checks that is polynomial in $|\mathcal{D}|$ (i.e. in the number of transactions), as follows. It requires first to build the inclusion graph for all the transactions in the dataset, with $O\left(|\mathcal{D}|^2\right)$ subgraph isomorphism checks (which cannot be done in polynomial time, rendering impossible to complete the task in time polynomial in the instance size unless P=NP). After this first step, for each possible value of d, we need to find the size of the largest anti-chain via a maximum matching problem on a suitably constructed bipartite graph with only the transactions of at least d nodes, which can be done in in time $O\left(\sqrt{|V|}|E|\right) = O\left(|\mathcal{D}|^{2.5}\right)$, e.g. using Dinic's algorithm. In fact, the correct value of d can be binary searched for a total complexity of $O\left(|\mathcal{D}|^{2.5} \log |\mathcal{D}|\right)$.This complexity makes it impossible to find the exact value

of the d-index on modern datasets, whose sizes easily exceed the millions of transactions.

We can obtain an upper bound to the d-index by computing the d-bound, that is the maximum number q such that there are at least q transactions with at least q nodes (i.e. disregarding the condition that such transactions must form an anti-chain), in a single scan of the dataset using Algorithm 1, which is a simple adaptation of the one presented in [RU14].

Algorithm 1: COMPUTED-BOUND($\mathcal{D}$)

```
1  q = 0
2  T = ∅
3  while hasNextTransaction(D) do
4  |  G = nextTransaction(D)
5  |  if |V_G| > q and ∄H ∈ T such that G = H then
6  |  |  T = T ∪ {G}
7  |  |  if ∀H ∈ T, |V_H| > q then
8  |  |  |  q++
9  |  |  else
10 |  |  |  remove from T the transaction H with minimum |V_H|
11 return q
```

Note that although a set of labeled graphs might form an anti-chain, their unlabeled counterparts might actually form some chains. Hence, labeled datasets should exhibit a slacker bound on the VC-dimension. Nonetheless, an important remark is that the definition of the d-bound and the corresponding bound on the VC-dimension of $(\mathcal{D}, R_{\mathcal{P}})$ are independent of the node and edge labels.

If we analyze the proof of Theorem 7, we can see that the condition $|V_{G_i}| \geq d$ is needed to upper bound the number of patterns in G_i. By making explicit this latter condition we can provide a stricter bound in the case where the pattern set $\mathcal{P}$ is small.

Definition 13. *Let $\mathcal{D}$ be a dataset. The c-bound of $\mathcal{D}$ w.r.t. $\mathcal{P}$ is the maximum integer c such that $\mathcal{D}$ contains at least c different transactions G_i such that G_i contains at least 2^{c-1} distinct patterns from $\mathcal{P}$.*

Theorem 8. *Let $\mathcal{D}$ be a dataset with c-bound c w.r.t the pattern set $\mathcal{P}$. Then the range space $(\mathcal{D}, R_{\mathcal{P}})$ has VC-dimension at most c.*

Proof. Let $l > c$ and assume that $(\mathcal{D}, R_{\mathcal{P}})$ has VC-dimension l. Then there is a set of $\mathcal{A}$ of l transactions of $\mathcal{D}$ that is shattered by $R_{\mathcal{P}}$, that is for each subset $\mathcal{B} \subseteq \mathcal{A}$, there exists a pattern $P_{\mathcal{B}}$ such that $T_{\mathcal{A}}(P_{\mathcal{B}}) = \mathcal{B}$. Note that there must be a graph $G \in \mathcal{A}$ that contains at most $2^{(c+1)-1} - 1$ patterns from $\mathcal{P}$, or $\mathcal{D}$ would have c-bound at least $c + 1$.

This graph G is a member of 2^{l-1} subsets of $\mathcal{A}$, which we call $\mathcal{A}_i$'s, labelled in any order. Since $\mathcal{A}$ is shattered by $R_{\mathcal{P}}$, we have that for each set $\mathcal{A}_i$, there exists a pattern P_i such that $T(P_i) \cap \mathcal{A} = \mathcal{A}_i$. Note that since the $\mathcal{A}_i$'s are all different, then also the P_i's must be all different. Since $G \in \mathcal{A}_i$ by construction, we must have that $G \in T(P_i)$ $\forall 1 \leq i \leq 2^{l-1}$. Then all such 2^{l-1} distinct patterns P_i appear as an induced subgraph of G. But, since $l > c$, we have that $2^{l-1} \geq 2^c > 2^c - 1$, and we have a contradiction. □

Note that the condition "G_i contains at least 2^{c-1} distinct patterns from $\mathcal{P}$" is much more difficult to test compared to the condition "G_i has at least d nodes". Hence, we provide a linear time algorithm to bound the c-bound of a dataset $\mathcal{D}$ in the particular case where $\mathcal{P}$ is the set of subgraphs with at most k nodes, which is a situation of interest in several subgraph mining applications.

Consider a graph $G \in \mathcal{D}$ with n_G nodes. Then G can contain at most $\hat{n}_G = \sum_{j=1}^{\min(k,n_G)} \min\left\{\binom{n_G}{j}, |\mathcal{P}_j|\right\}$ patterns from $\mathcal{P}$, where $|\mathcal{P}_j|$ is the number of patterns in $\mathcal{P}$ with j nodes.

Note that often $\sum_{j=1}^{\min(k,n)} \binom{n}{j} \ll 2^n$, making this bound much tighter than the d-bound. Also, note that in this case $|\mathcal{P}_j|$, although difficult to compute in closed-form, can be precomputed using values from `oeis.org/A001349`. Especially in the case of labelled graphs, though, $|\mathcal{P}|$ grows very quickly with k and is rarely useful.

Let then $c_G = \lfloor \log_2(\hat{n}_G) \rfloor + 1$ and let $\hat{c}$ be the maximum integer c such that at least c transactions G have $c_G \geq c$. This value can be computed in a single scan of the dataset with a straightforward adaptation of Algorithm 1, which we report for completeness in Algorithm 2.

Algorithm 2: COMPUTEC-BOUND($\mathcal{D}$)

$q = 0$
$T = \emptyset$
while *hasNextTransaction*($\mathcal{D}$) **do**
  $G =$ NEXTTRANSACTION($\mathcal{D}$)
  $c_G = \left\lfloor \log \left(\sum_{j=1}^{\min(k,n_G)} \min \left\{ \binom{n_G}{j}, |\mathcal{P}_j| \right\} \right) \right\rfloor + 1$
  if $c_G > q$ *and* $\nexists H \in T$ *such that* $G = H$ **then**
    $T = T \cup \{G\}$
    if $\forall H \in T,\ c_H > q$ **then**
      $q++$
    else
      remove from T the transaction H with minimum c_H
return q

Note that, if the c-bound of $\mathcal{D}$ is c^*, then $c^* \leq \hat{c}$. Indeed, consider c^* graphs such that each one of them has at least 2^{c^*-1} distinct patterns from $\mathcal{P}$. Then for each of such graphs we have $2^{c^*-1} \leq \hat{n}_G$, so $c_G \geq c^*$. Then, since there are at least c^* graphs such that each one has $c_G \geq c^*$, we have $\hat{c} \geq c^*$. Hence, computing $\hat{c}$ provides an upper bound to the VC-dimension of $(\mathcal{D}, R_{\mathcal{P}})$.

Chapter 4

Rademacher Averages of Subgraphs

We now define Rademacher averages for subgraph patterns, specializing the definitions of Section 2.3 to the subgraph mining problem.

For each pattern (i.e. subgraph) $P \in \mathcal{P}$, define the indicator function $\phi_P : \mathcal{D} \to \{0, 1\}$ as

$$\phi_P(G) = \begin{cases} 1 \text{ if } P \subseteq G \\ 0 \text{ otherwise.} \end{cases}$$

Then we have $f_{\mathcal{D}}(P) = \frac{1}{|\mathcal{D}|} \sum_{G \in \mathcal{D}} \phi_P(G)$.

Assume to take a random sample (uniformly and with replacement) $\mathcal{S} = \{G_1, \ldots, G_n\}$ of size n from $\mathcal{D}$. Then let σ_i be a Rademacher random variable, i.e. taking value 1 or -1 with probability $1/2$, for each $1 \leq i \leq n$. Let also the σ_i's be independent. Then we define the empirical Rademacher average in the context of frequent subgraph mining as

$$\mathcal{R}_{\mathcal{S}} = \mathbb{E}_{\sigma} \left[\sup_{P \in \mathcal{P}} \frac{1}{n} \sum_{i=1}^{n} \sigma_i \phi_P(G_i) \right]$$

where $\mathbb{E}_\sigma$ denotes the expectation taken only with respect to the σ_i's, conditionally on the sample.

4.1 Bounding the Rademacher Average

In this section we show how to efficiently bound $\mathcal{R}_\mathcal{S}$. Indeed, note that computing $\mathcal{R}_\mathcal{S}$ directly is infeasible, as it involves computing the support of all patterns, and we hence resort to bounds given by Massart's lemma, which is the most widely used tool to bound Rademacher averages without resorting to time-consuming Monte Carlo methods, as described in Section 2.3.

For any pattern $P \in \mathcal{P}$, let $v_\mathcal{S}(P)$ be the n-dimensional binary vector $(\phi_P(G_1), \ldots, \phi_P(G_n))^T$. Then, let $V_\mathcal{S} = \{v_\mathcal{S}(P),\ P \in \mathcal{P}\}$. Since $V_\mathcal{S}$ is a set, we may have $|V_\mathcal{S}| \ll |\mathcal{P}|$ in case many patterns exhibit the support in $\mathcal{S}$.

Theorem 9 (Massart's Lemma, frequent subgraph mining formulation)**.**

$$\mathcal{R}_\mathcal{S} \leq \max_{P \in \mathcal{P}} \|v_\mathcal{S}(P)\| \frac{\sqrt{2 \ln |V_\mathcal{S}|}}{n} = \max_{P \in \mathcal{P}} \sqrt{\frac{2 f_\mathcal{S}(P) \ln |V_\mathcal{S}|}{n}}.$$

Actually, we can use the stronger version of the theorem, as formulated in Lemma 3, as follows.

Theorem 10. *Let $w : \mathbb{R}^+ \to \mathbb{R}^+$ be the function*

$$w(s) = \frac{1}{s} \ln \left(\sum_{v \in V_\mathcal{S}} \exp\left(s^2 \|v\|^2 / (2n^2)\right) \right).$$

Then $\mathcal{R}_\mathcal{S} \leq \min_{s \in \mathbb{R}^+} w(s)$.

Once again, computing w is infeasible as it would involve computing all supports. We then devise a function $\tilde{w}$, computable with a single scan of the sample, that upper bounds w and can hence be used to bound the Rademacher average.

Observation 1. *An important observation is that if we restrict $\mathcal{P}$ to the set of patterns with strictly positive frequency $\mathcal{P}^+$, the maximum deviation of the*

frequency is unchanged. Hence, we can work with the set of all connected induced subgraphs of all the transactions, which is a huge but finite set, rather that the infinite set of all possible connected graphs.

In the following paragraphs we show how to compute efficiently an upper bound to the Rademacher average in different scenarios.

4.1.1 Labeled Graphs

In this section we show how to efficiently bound $\mathcal{R}_{\mathcal{S}}$ for datasets of node labeled graphs.

Note that we can see the individual labeled nodes as the building blocks of the transactions. Let L be the set of such labeled nodes. It is useful to consider them as patterns, as it will help to transition from labeled to unlabeled graphs.

We partition $V_{\mathcal{S}}$ as follows. Let $C_i = \{P = (V_P, E_P) \in \mathcal{P}^+ \text{ s.t. } |V_P| = i\}$ be the set of patterns with i nodes and $V_i = \{v_P \ : \ P \in C_i\}$.

Then, consider the following partitioning of the sets V_i, $i \geq 1$. Assume to sort the labels in increasing order by the frequency of their corresponding pattern in $\mathcal{S}$ and let the resulting order be $<_\ell$ (other orderings will work as well, but might lead to worse bounds). Let $v_P \in V_i$ and let $Q \in L$ be s.t. Q is the first pattern in the ordering $<_\ell$ such that $Q \subseteq P$. Note that there always exists one such pattern, since transactions are non-empty. Then we assign v_P to the set $V_{i,Q}$.

Let $T_{\mathcal{S}}(i, Q)$ be the set of transactions $\tau = (V_\tau, E_\tau) \in \mathcal{S}$ s.t. $Q \subseteq \tau$ and at least i nodes have label $\geq_\ell Q$. Consider then a transaction $\tau = (V_\tau, E_\tau) \in T_{\mathcal{S}}(i, Q)$. Let $M_{i,Q,\tau}$ be the set of all induced connected subgraphs H of τ s.t. H has i vertices, it contains pattern Q, and it does not contain patterns $Q' <_\ell Q$. Let also $m_{i,Q,\tau} = |M_{i,Q,\tau}|$. We then have the following lemma.

Lemma 6. *We have* $|V_{i,Q}| \leq \sum_{\tau \in T_{\mathcal{S}}(i,Q)} m_{i,Q,\tau}$.

Proof. We show that there exists an injective function $f : V_{i,Q} \longrightarrow \bigcup_{\tau \in T_{\mathcal{S}}(i,Q)} M_{i,Q,\tau}$. Let $v_P \in V_{i,Q}$, and first of all note that P cannot contain nodes $Q' <_\ell Q$, as v_P would belong to $V_{i,Q'}$. Then the pattern P, since

it has positive frequency, is isomorphic to a connected induced subgraph H of some $\tau \in \mathcal{S}$, with τ containing at least a node Q and at least i nodes with label $\geq_\ell Q$. Then $H \in M_{i,Q,\tau},\ \tau \in T_{\mathcal{S}}(i,Q)$. Let then $f(v_P) = H$. In case of multiple possible H's, ties are broken arbitrarily. Since P and $f(v_P)$ are isomorphic, no two different patterns can be mapped to the same connected induced subgraph H, and f is hence injective. Recalling that $|\bigcup_{\tau \in T_{\mathcal{S}}(i,Q)} M_{i,Q,\tau}| \leq \sum_{\tau \in T_{\mathcal{S}}(i,Q)} m_{i,Q,\tau}$, we have the claim. □

A slack bound to $m_{i,Q,\tau}$ is $\binom{|V_\tau|}{i}$. A better bound can be obtained noticing that nodes $Q' <_\ell Q$ cannot be chosen, and that at least one node Q must be chosen. Let $|V_\tau^{(\geq Q)}|$ be the number of nodes in τ with label $\geq_\ell Q$ and $|V_\tau^{(>Q)}|$ be the number of nodes with label $>_\ell Q$. Then we have $m_{i,Q,\tau} \leq \binom{|V_\tau^{(\geq Q)}|}{i} - \binom{|V_\tau^{(>Q)}|}{i} \leq \binom{|V_\tau^{(\geq Q)}|}{i}$.

Note that in the bounds above we are not exploiting the fact that the subgraphs have to be connected. Then, if one is willing to sacrifice running times for obtaining better bounds, an exact subgraph enumeration algorithm can be suitably modified to provide a tighter bound to $m_{i,Q,\tau}$. We describe such an algorithm in Section 4.2.

We now provide a second bound on $|V_{i,Q}|$, which exploits the fact that many patterns might share the same support.

Lemma 7. *We have* $|V_{i,Q}| \leq 2^{|T_{\mathcal{S}}(i,Q)|} - 1$.

Proof. Note that all transactions in $\mathcal{S} \setminus T_{\mathcal{S}}(i,Q)$ must have the corresponding entry in v_P set to 0 for all patterns $P \in V_{i,Q}$. Indeed, if a transaction has less than i nodes with label $\geq_\ell Q$ it cannot contain P, as it has i nodes and does not contain any node with label $<_\ell Q$. Moreover, since Q is an induced subgraph of P it is impossible for a transaction to contain P but not Q. Then there are only $2^{|T_{\mathcal{S}}(i,Q)|}$ possible assignments, one of which is of all zeros. □

Lemma 8. *Define* $m_{i,Q} = \sum_{\tau \in T_{\mathcal{S}}(i,Q)} m_{i,Q,\tau}$ *and* $m'_{i,Q} = 2^{|T_{\mathcal{S}}(i,Q)|} - 1$. *Let also* χ *be the maximum number of nodes in any transaction. Let then* $\tilde{w} : \mathbb{R}^+ \to \mathbb{R}^+$ *be the function*

$$\tilde{w}(s) = \frac{1}{s} \ln \left(\sum_{i=1}^{\chi} \sum_{Q \in L} \min\{m_{i,Q}, m'_{i,Q}\} e^{\frac{s^2 |T_{\mathcal{S}}(i,Q)|}{2n^2}} \right).$$

Then we have that $w(s) \leq \tilde{w}(s)$, $\forall s \in \mathbb{R}$.

Proof. We can write $w(s)$ as follows.

$$w(s) = \frac{1}{s} \ln \left(\sum_{v_P \in V_{\mathcal{S}}} \exp\left(s^2 \left\|v_P\right\|^2 / (2n^2)\right) \right)$$
$$= \frac{1}{s} \ln \left(\sum_{i=1}^{\chi} \sum_{Q \in L} \sum_{v \in V_{i,Q}} e^{\frac{s^2 \|v\|^2}{2n^2}} \right).$$

Moreover, as shown in Lemma 7, all transactions in $\mathcal{S} \setminus T_{\mathcal{S}}(i,Q)$ must have the corresponding entry in v_P set to 0 for all patterns $P \in V_{i,Q}$. Then for such patterns we have $\left\|v_P\right\|^2 \leq |T_{\mathcal{S}}(i,Q)|$. Then we have

$$\sum_{v \in V_{i,Q}} e^{\frac{s^2 \|v\|^2}{2n^2}} \leq e^{\frac{s^2 |T_{\mathcal{S}}(i,Q)|}{2n^2}} \cdot |V_{i,Q}|.$$

Combining the bounds on $|V_{i,Q}|$ from Lemmas 6 and 7 we obtain the claim. □

Note that, if one is interested in bounding the deviations of frequencies only for subgraphs of size up to k, one can simply put $\chi = k$ and obtain a tighter number of samples.

Observation 2. *By design, the two quantities* $m_{i,Q}$ *and* $m'_{i,Q}$ *are useful in different transaction size ranges. Indeed, the former is more powerful for small node counts* i*, where the cardinality of* $T_{\mathcal{S}}(i,Q)$ *is very large and thus not useful, while the number of connected subgraphs of size* i *is relatively small. For large node counts* i*, instead, the number of connected subgraphs is huge, but the size of* $T_{\mathcal{S}}(i,Q)$ *is much smaller, as large transactions are hopefully rare.*

We can compute all the needed quantities in a single scan of the sample $\mathcal{S}$, provided that the order $<_\ell$ is already available. Indeed, for each $\tau = (V_\tau, E_\tau) \in \mathcal{S}$, we check for the presence of patterns $P \in L$ in τ. Then, for each $Q \in L$ in increasing order by $<_\ell$, if $Q \subseteq \tau$, we upper bound the quantity $m_{i,Q,\tau}$, either using the binomial formula or a subgraph enumeration algorithm. Moreover, we update the size of $T_{\mathcal{S}}(i,Q)$ for each $i = 1, \ldots, |V_\tau^{(\geq_\ell Q)}|$.

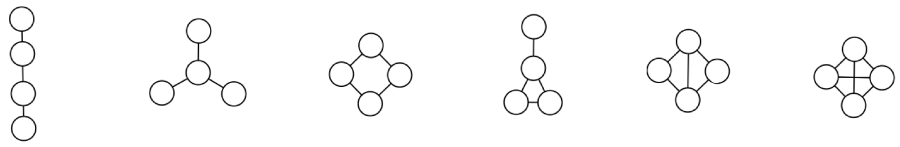

Figure 4.1: 4-node connected subgraphs

Then, we can minimize $\tilde{w}(s)$ using a nonlinear optimization solver to obtain a bound to $\mathcal{R}_{\mathcal{S}}$. The pseudocode of the procedure is provided in Algorithm 3.

Algorithm 3: EstimateRs($\mathcal{S}$)

$m_{i,Q} = 0 \ \forall i, \ \forall Q \in L$
for $\tau \in \mathcal{S}$ **do**
  for $Q \in L$ **do**
    if $Q \subseteq \tau$ **then**
      $m_{i,Q} += \binom{|V_\tau^{(\geq Q)}|}{i} - \binom{|V_\tau^{(>Q)}|}{i}$
      for $i = 1, \ldots, |V_\tau^{(\geq Q)}|$ **do**
        $T_{i,Q}++$
$m'_{i,Q} = 2^{T_{i,Q}} - 1, \ \forall i, \ \forall Q \in L$
$\tilde{w}(s) = \frac{1}{s} \ln \left(\sum_{i=1}^{\chi} \sum_{Q \in L} \min\{m_{i,Q}, m'_{i,Q}\} e^{\frac{s^2 T_{i,Q}}{2n^2}} \right)$
return $\min_{s \in \mathbb{R}^+} \tilde{w}(s)$

4.1.2 Unlabeled Graphs

The method described above still holds for unlabeled graphs, as we can treat all nodes as having the same (empty) label. Of course though this weakens the bound, as we are really only partitioning $V_{\mathcal{S}}$ on the number of nodes of the patterns.

Rather than considering the single nodes as the building blocks of the transactions, we can consider some small subgraphs of size $k = 4$ or 5 instead (e.g. the graphs of Fig. 4.1). Then, we will compute exactly the frequency for all patterns of size up to k, and use the following partitioning for the sets V_i,

$i > k$. Assume again to sort patterns of C_k according to some ordering $<_k$ (e.g. by frequency). Let $v_P \in V_i$ and let $Q \in C_k$ be s.t. Q is the first pattern in the ordering $<_k$ such that $Q \subseteq P$. Then we assign v_P to the set $V_{i,Q}$.

Note that, since in the labeled case we treated individual nodes as patterns, most of the analysis carries over to this case trivially, with the exception of the bounds to $m_{i,Q,\tau}$.

Indeed, while it is still true that $m_{i,Q,\tau} \leq \binom{|V_\tau|}{i}$, it is not possible to exploit the fact that patterns $Q' <_k Q$ should not be chosen to obtain a closed form bound. Similarly, a subgraph enumeration procedure should be adapted as well.

We then have the following lemma, whose proof is akin to the one of Lemma 8:

Lemma 9. *Let $m_{i,Q}$, $m'_{i,Q}$ and χ be defined as before. Let then $\tilde{w} : \mathbb{R}^+ \to \mathbb{R}^+$ be the function*

$$\tilde{w}(s) = \frac{1}{s} \ln \left(\sum_{i=1}^{k} \sum_{P \in C_i} e^{\frac{s^2 |T_{\mathcal{S}}(Q)|}{2n^n}} + \sum_{i=k+1}^{\chi} \sum_{Q \in C_k} \min\{m_{i,Q}, m'_{i,Q}\} e^{\frac{s^2 |T_{\mathcal{S}}(i,Q)|}{2n^2}} \right).$$

Then we have that $w(s) \leq \tilde{w}(s)$, $\forall s \in \mathbb{R}$.

4.2 Bounding the Number of Connected Subgraphs

As stated before, bounding $m_{i,Q,\tau}$ with the binomial formula yields quite a slack bound, since we are considering also disconnected graphs. We now briefely describe a subgraph enumeration algorithm which only considers connected graphs.

Given a graph τ, we perform a complete search incrementally growing an induced subgraph *cur* by adding to it vertices from the ones we haven't considered yet (which we store in $left$), intersected with the neighbors of the

vertices in *cur*, which we store in *neigh*. Initially, we start with *cur* and *neigh* empty and with $left$ the set of nodes that can be used, namely the ones with label $\geq_\ell Q$. To contain running times, we limit the recursion depth, using the simpler bounds on nodes in $left$ for the subtrees below the maximum depth.

Algorithm 4: CountSubgraphs($cur, left, neigh, Q, remDepth$)

```
1  if remDepth == 0 then
2  |   cnt_cur = # nodes in cur
3  |   cnt_left = # nodes in left
4  |   for i = 0, ..., cnt_left do
5  |   |   m_{(cnt_cur+i),Q,τ} += (cnt_left choose i)
6  |   return
7  if cur == ∅ then
8  |   cand = left
9  else
10 |   cand = left ∩ neigh
11 if cand ≠ ∅ then
12 |   pick first v from cand
13 |   CountSubgraphs(cur, left \ {v}, neigh, Q, remDepth − 1)
14 |   CountSubgraphs(cur∪{v}, left\{v}, neigh∪N(v), Q, remDepth−1)
15 else
16 |   cnt_cur = # nodes in cur
17 |   m_{cnt_cur,Q,τ} += 1
```

It's easy to see that, if we do not limit the recursion depth, all and only the connected induced subgraphs of τ are counted in lines 15-17. Indeed, consider a connected induced subgraph $G = (V = \{i_1, \ldots, i_l\}, E)$. Then i_1 will be picked from *cand* and added to *cur* in the branch of the recursion tree where all previous nodes were not added to *cur*. Afterwards, all vertices $i_2, \ldots, i_l$ will be added to *cand* at some point since they are connected (either directly or indirectly) to i_1. Hence, G will be created as a leaf of the recursion tree. Conversely, a disconnected subgraph cannot be created since once the first vertex is added to *cur*, the nodes in the other connected components cannot be added to *cand*.

At each internal node of the recursion tree we perform $O(|V|)$ work (very fast if implemented using bitmaps) and in the leaves we perform $O(|V|)$ work as well. Then, since in a binary tree with k leaves we have $k-1$ internal

nodes, called z the number of connected induced subgraphs of τ, we have a total time complexity of $O\left(z \cdot |V|\right)$.

Chapter 5

Applications

This chapter describes two applications of the bounds on the VC-dimension and on Rademacher averages for subgraphs. In particular, Section 5.1 describes their use for approximate frequent subgraph mining through sampling, while Section 5.2 describes their use to solve the problem of true frequent subgraph mining.

5.1 Approximate Frequent Subgraph Mining with Guarantees

We now propose two algorithms, one based on the VC-dimension and one based on Radamacher Averages, to compute ε-approximations of frequent subgraphs by sampling. As with all sampling approaches, the main challenge is to bound the number of samples required to have guarantees on the relation between the results on the sample and the results on the whole dataset. In particular, we are interested in obtaining ε-close approximations.

As shown in Section 2.2, a straightforward application of Hoeffding's inequality yields that, for a fixed pattern P, if we take a sample $\mathcal{D}'$ of $\frac{1}{2\varepsilon^2}\ln(\frac{2}{\delta})$ transactions, with probability $1-\delta$ we have that $|f_{\mathcal{D}'}(P) - f_{\mathcal{D}}(P)| \leq \varepsilon$. One generally would then proceed to bound the probability of not obtaining a ε-close approximations using union bound over all patterns. In the case of frequent subgraph mining though this strategy fails due to the enormous number of patterns in $\mathcal{P}$, which is exponential in the square of the maximum number of nodes in a pattern. We then exploit the bounds we derived in the previous sections to develop better methods.

5.1.1 VC-dimension-based Algorithm

We now present our algorithm based on the VC-dimension. In particular, the algorithm first computes the upper bound to the VC-dimension using Algorithm 2. Then, it takes a random sample $\mathcal{D}'$, taken uniformly at random, of $\mathcal{D}$ and reports in output $FG(\mathcal{D}', \mathcal{P}, \theta - \varepsilon)$, that is, all subgraphs with frequency $\geq \theta - \varepsilon$ in $\mathcal{D}'$. The following theorem (whose proof is analogous to the one in [RU14] for frequent itemsets mining) bounds on the number of samples that $\mathcal{D}'$ must contain for the output $FG(\mathcal{D}', \mathcal{P}, \theta - \varepsilon)$ to be a 2ε-close approximation of $FG(\mathcal{D}, \mathcal{P}, \theta)$ with probability at least $1-\delta$.

Theorem 11. *Let $\mathcal{D}$ be a dataset of transactions, $\mathcal{P}$ a set of patterns and v an upper bound to the VC dimension of the range space associated with them. Let $0 < \varepsilon, \delta < 1$. Let $\mathcal{D}'$ be a random sample of $\mathcal{D}$ (taken uniformly at random with replacements) of size*

$$\min\left\{|\mathcal{D}|,\ \frac{c}{\varepsilon^2}\left(v + \ln\frac{1}{\delta}\right)\right\}$$

for some absolute constant c. If $m = |\mathcal{D}|$ we assume $\mathcal{D}' = \mathcal{D}$. Then $FG(\mathcal{D}', \mathcal{P}, \theta - \varepsilon)$ is an absolute (2ε)-close approximation to $FG(\mathcal{D}, \mathcal{P}, \theta)$ with probability at least $1-\delta$.

Proof. From Theorem 2 we know that with probability at least $1-\delta$, $\mathcal{D}'$ is an ε-approximation for $\mathcal{D}$, i.e. $|f_{\mathcal{D}'}(P) - f_{\mathcal{D}}(P)| \leq \varepsilon$ for each $P \in \mathcal{P}$. In

particular, this happens for all $P \in C = FG(\mathcal{D}', \mathcal{P}, \theta - \varepsilon)$, satisfying property 3 of Definition 2. Moreover, for each $P \in FG(\mathcal{D}, \mathcal{P}, \theta)$, we have $f_{\mathcal{D}}(P) \geq \theta - \varepsilon$, so it appears in C, satisfying property 1. Finally, if P is s.t. $f_{\mathcal{D}}(P) < \theta - 2\varepsilon$, then $f_{\mathcal{D}}(P) < \theta - \varepsilon$ and it does not appear in C, satisfying property 2. □

5.1.2 Rademacher-average-based Algorithm

As a second approach for obtaining ε-close approximations with guarantees is to use a progressive sampling scheme [RU15], with Rademacher averages to define stopping conditions. This avoids to process the entire dataset, which can be beneficial in extremely massive datasets, allowing to consider only a small sample of it. The outline of the sampling algorithm is the following:

1. at iteration i, obtain the random sample $\mathcal{D}_i$ from $\mathcal{D}$;
2. compute ε_R such that $\max_{P \in \mathcal{P}} |f_{\mathcal{D}_i}(P) - f_{\mathcal{D}}(P)| \leq \varepsilon_R$;
3. check if $\varepsilon_R \leq \varepsilon$;
4. if so, return $\mathcal{D}' = \mathcal{D}_i$, else compute the sample size at iteration $i + 1$, increase i and return to (1).

The computation of ε_R is performed using Rademacher averages. Indeed, we have the following result, based on Theorem 4, bounding the maximum deviation of pattern frequencies.

Theorem 12. *Let $\mathcal{S}$ be a random sample (uniformly and with replacements) of size n from $\mathcal{D}$. Let also $\mathcal{R}_{\mathcal{S}}$ be the empirical Rademacher average of $\mathcal{S}$, as defined in Chapter 4. Then, with probability $1 - \delta$,*

$$\sup_{P \in \mathcal{P}} |f_{\mathcal{D}}(G) - f_{\mathcal{S}}(G)| \leq 2\mathcal{R}_{\mathcal{S}} + 3\sqrt{\frac{\ln 2/\delta}{2n}}.$$

In particular, Lemma 8 (or Lemma 9, if the graphs are unlabelled) is used to compute an upper bound $\tilde{w}(s)$ to $w(s)$ for each $s \in \mathbb{R}$, which leads to an upper bound to the Rademacher average $\mathcal{R}_{\mathcal{S}}$ using Theorem 10. Such upper bound to $\mathcal{R}_{\mathcal{S}}$ is used to compute the probabilistic upper bound ε_R

to $\max_{P \in \mathcal{P}} |f_{\mathcal{D}'}(P) - f_{\mathcal{D}}(P)|$ according to Theorem 12, which holds with probability at least $1-\delta$. Then, using the arguments employed in Theorem 11, we can prove that $FG(\mathcal{D}', \mathcal{P}, \theta-\varepsilon)$ is a 2ε-close approximation for $FG(\mathcal{D}, \mathcal{P}, \theta)$.

The last component in the progressive sampling algorithm to be described is the choice of a sampling schedule, which is defined by the initial sample size and by the growth rate of the number of samples.

As shown in [RU15], if the number of samples is smaller than $S_0^* = \frac{9 \ln(2/\delta)}{2\varepsilon^2}$, then the condition of Theorem 12 cannot be satisfied, so we choose S_0^* as the initial sample size.

As for the growth rate, one possibility would be to choose a geometric sampling schedule $|S_{i+1}| = \alpha |S_i|$, with α a parameter to be chosen by the user. In fact, as discussed in [RU15], a better sampling strategy is to use the estimated maximum deviation at iteration i, $\varepsilon_R = \min_s \tilde{w}(s) + 3\sqrt{\frac{\ln(2/\delta)}{2|S_i|}}$, to generate the next sample size. Indeed, since there is a quadratic dependency between the sample size and the maximum deviation, a good guess for the next sample size is $|S_{i+1}| = \left(\frac{\varepsilon_R}{\varepsilon}\right)^2 |S_i|$.

5.2 True Frequent Subgraph Mining

As described in Section 2.1, a common scenario in frequent pattern mining is that the dataset $\mathcal{D}$ is not to be considered as the ground truth on the underlying generating process, but rather as a sample from an unknown generating distribution π. In this setting then one would like to extrapolate, from the sample $\mathcal{D}$, the patterns that are frequent in the underlying distribution. More formally, as defined in Section 2.1, the *true frequent patterns* are all the patterns P such that $\sum_{G:P \subseteq G} p_\pi(G) \geq \theta$, where p_π is the density function of π.

A simple algorithm to compute a 2ε-close approximation of the set of true frequent subgraphs with probability at least $1-\delta$ is analogous to the one proposed in [RV14] for true frequent itemset mining, and works as follows: using dataset $\mathcal{D}$, compute an upper bound ε to $\max_{P \in \mathcal{P}} |p_\pi(P) - f_{\mathcal{D}}(P)|$ that holds with probability at least $1-\delta$; report in output the set of patterns

$FG(\mathcal{D}, \mathcal{P}, \theta - \varepsilon)$ with frequency at least $\theta - \varepsilon$ in the dataset $\mathcal{D}$. The output is then a 2ε-close approximation of the set of true frequent subgraphs with probability at least $1 - \delta$ (the proof is analogous to the one in [RV14]).

The upper bound ε to $\max_{P \in \mathcal{P}} |p_\pi(P) - f_\mathcal{D}(P)|$ that holds with probability at least $1 - \delta$ can be computed using either the VC-dimension (see Chapter 3) or Rademacher averages (see Chapter 4), as described in the following subsections.

5.2.1 Bounding Deviations using the Empirical VC Dimension

In this context we have to use the more general notion of ε-approximation we introduced in Definition 9. While the results of Theorem 2 hold even for the general definition of ε-approximations that we are using in this context, albeit with the caveat that sampling has to be performed according to π rather than uniformly, it is impossible to bound the VC dimension of the range space $(\Pi, R_\mathcal{P})$ associated with π directly, as the support Π of π is in general unknown. We then have to resort to a weaker bound based on the empirical VC dimension on the sample $\mathcal{D}$. Indeed, if we let $d = EVC((\Pi, R_\mathcal{P}), \mathcal{D}) = VC(\mathcal{D}, R_\mathcal{P})$, thanks to Lemma 4, we have that the sample $\mathcal{D}$, of size n, is an ε-approximation for Π w.r.t. π for

$$\varepsilon = 2\sqrt{\frac{2d\ln(n+1)}{n}} + 3\sqrt{\frac{\ln(2/\delta)}{2n}}.$$

The VC dimension of $(\mathcal{D}, R_\mathcal{P})$ can be bounded using either the d-bound or the c-bound, as described in Chapter 3, giving then a bound on the empirical VC dimension of $(\Pi, R_\mathcal{P})$ on $\mathcal{D}$. This immediately yields a bound on the maximum deviation of the frequencies of patterns in $\mathcal{D}$ with respect to their true frequencies in the generating distribution π. This can be in turn used to obtain absolute (2ε)-close approximations to the true frequent subgraph mining problem.

5.2.2 Bounding Deviations using Rademacher Averages

While the bounds based on the VC dimension required information on the entire range space of the instance, the Rademacher-averages-based bounds require only *sample dependent* quantities, and can thus be directly used to estimate the maximum deviation of frequencies even when the dataset $\mathcal{D}$ is to be considered as a sample from an unknown generating distribution. In Chapter 6 we provide experimental evidence that this approach significantly outperforms the method based on the empirical VC dimension.

Chapter 6

Experiments

In order to evaluate the effectiveness of our theoretical bounds, we implemented the methods to compute the sample sizes described in this thesis and tested them on a suite of proof-of-concept experiments. The main goals of this experimental evaluation is to

- Compare the tightness of the bounds we presented both in the *sampling from a static dataset* and in the *sampling from a distribution* version of the problem
- Assess the precision, recall and maximum deviation of the approximate mining algorithms derived from our sampling schemes, for both versions of the problem.
- When sampling from a fixed dataset, show the performance benefits in terms of running time and peak memory consumption

All experiments were performed on a server with a dual Intel Xeon 5220 processor with 72 cores and 1Tb of RAM. We implemented our algorithms

Table 6.1: Key properties of datasets. See Section 2.1 for the definition of the properties.

Name	$\lvert\mathcal{D}\rvert$	$\lvert L_V\rvert$	$\lvert L_E\rvert$	avg. $\lvert V\rvert$	avg. $\lvert E\rvert$
Akos	10000000	76	2	49.6	51.6
Pubchem	25000000	103	2	50.1	52.1
Reddit	203088	1	1	23.9	24.9
Alphafold	541374	20	3	353.8	373.8

in C++ and used no form of paralellism. Moreover, in order to perform the subgraph isomorphism checks within our methods, we have implemented a wrapper around the VF3Lib library [CFSV18], as it is reportedly one of the fastest ones. As the nonlinear optimizer, we used the NLopt library [NLo]. For what concerns the subgraph mining step, we used the Gaston library [NK05], as again it is one of the most performant ones [WMFP05]. This mining algorithm has two versions, one with occurrence lists, which is fast but uses a significant amount of memory, and one without occurrence lists (Gaston RE) that uses a small amount of memory but is significantly slower that the other version. For labelled datasets, where the search space can be efficiently pruned, we were able to use the former version, while for unlabelled datasets the mining process on Gaston with occurrence lists exceeded one terabyte of main memory even for small subgraph sizes and high frequency thresholds, so we had to resort to the RE version of the miner.

6.1 Datasets

We used datasets from various fields, including computational chemistry, computational biology and social network analysis. The Akos and Pubchem datasets are subsets of two well-known molecular databases[1], and have already been used for validating large-scale subgraph mining [LXG14]. Reddit[2] is a collection of graphs representing threads collected from Reddit in May

[1] http://akosgmbh.de/, https://pubchem.ncbi.nlm.nih.gov
[2] https://snap.stanford.edu/data

2018 and is the only unlabeled dataset. Moreover, we introduce a new graph dataset ALPHAFOLD based on the protein structure predictions of DeepMind's Alphafold on the SwissProt dataset[3]. Since many proteins feature a high node count, this dataset can be considered a stress test for our methods, as their performance should degrade as the average size of the graphs grows.

Table 6.1 enumerates the datasets as well as some of their key properties.

Most subgraph miners (e.g. Gaston [NK05] and gSpan [YH02]) use the following format for encoding graph transactional datasets:

```
t # <graph-id>
v <vertex-id> <vertex-label>
...
e <vertex-id> <vertex-id> <edge-label>
...
```

We then had to convert the datasets from the format they were encoded in to the format above. The next subsections describe the preprocessing steps that that had to be performed for each dataset.

6.1.1 Molecular Datasets Preprocessing

The AKOS and PUBCHEM databases can be dowloaded in SDF format, which encodes information about the atoms, bonds, and coordinates of the atoms of a molecule. First of all, since we only need connectivity information (and not spatial information), we can efficiently store each molecule into its SMILES string, which is a compact way of describing molecular graphs, saving a significant amount of space. To do so we used the RDKit[4] library in Python, which is widely spread in computational chemistry. The same library allows us to transform the molecules into the graph format we described above.

The following is a snipped of the code used to produce the transactional datasets.

[3]https://alphafold.ebi.ac.uk/
[4]https://www.rdkit.org/

```
import sys
from rdkit import Chem

count = 0

def converter(file_name):
  global count
  sppl = Chem.SDMolSupplier(file_name)
  smiles_file = open(out_path+"_smiles.txt", "w")
  graphs_file = open(out_path+"_graphs.txt", "w")
  for mol in sppl:
    if mol is not None:# some compounds cannot be loaded.
      smi = Chem.MolToSmiles(mol,
                             allHsExplicit = True,
                             allBondsExplicit = True)
      smiles_file.write(f"{smi}\n")

      atoms_info = [ (atom.GetIdx(), atom.GetAtomicNum())
                     for atom in mol.GetAtoms(onlyHeavy=False)]
      bonds_info = [(bond.GetBeginAtomIdx(),
                     bond.GetEndAtomIdx(),
                     int(bond.GetBondTypeAsDouble()*2 - 2))
                    for bond in mol.GetBonds()]
      graphs_file.write(f"t␣#␣{count}\n")
      for (id, lb) in atoms_info:
        graphs_file.write(f"v␣{id}␣{lb}\n")
      for (u, v, lb) in bonds_info:
          graphs_file.write(f"e␣{u}␣{v}␣{lb}\n")

      count += 1
  print("Converted", count, "molecules")
  smiles_file.close()
  graphs_file.close()
```

6.1.2 Protein Dataset Preprocessing

DeepMind provides its database on the protein structures predicted by AlphaFold [JEP+21] on the SwissProt protein dataset. The protein structures are stored in the PDB format. We generated the graphs, using the Graphein[5] library, using the amino acids as nodes and peptide, hydrogen and aromatic bonds as edges. The graphs obtained form this procedure are stores using the NetworkX library[6], which is easy to interface with Graphein. Once the graphs were generated, we transformed them into the standard format we decribed above.

[5]https://graphein.ai/
[6]https://networkx.org/

The following is a snipped of the code used to produce the ALPHFOLD transactional dataset.

```
from graphein.protein.config import ProteinGraphConfig
from graphein.protein.graphs import construct_graph
from graphein.protein.edges.distance import add_hydrogen_bond_interactions,
                                            add_aromatic_interactions,
                                            add_peptide_bonds
from graphein.protein.features.nodes.amino_acid import amino_acid_one_hot

new_funcs = {"edge_construction_functions":
             [add_peptide_bonds,
              add_aromatic_interactions,
              add_hydrogen_bond_interactions],
             "node_metadata_functions":
             [amino_acid_one_hot]
             }
config = ProteinGraphConfig(**new_funcs)

count = 0

for filename in os.listdir(inpath):
  filepath = os.path.join(inpath, filename)
  if(not filename.endswith(".pdb.gz")):
    continue
  g = construct_graph(config=config, pdb_path=filepath)
  names2id = {}
  cnt = 0
  for n, d in g.nodes(data=True):
    names2id[n] = cnt
    cnt += 1
  out.write(f"t␣#␣{count}\n")
  for n, d in g.nodes(data=True):
    out.write(f"v␣{names2id[n]}␣{np.argmax(d['amino_acid_one_hot'])}\n")
  for n1, n2, d in g.edges(data=True):
    dd = 0
    if 'hbond' in d['kind']:
      dd = 1
    if 'aromatic' in d['kind']:
       dd = 2
    if 'peptide_bond' in d['kind']:
      dd = 0
    out.write(f"e␣{names2id[n1]}␣{names2id[n2]}␣{dd}\n")
  out.flush()
  count += 1

out.close();
```

6.1.3 Reddit Dataset Preprocessing

The REDDIT dataset is provided in json format as an edge list, which can be easily transformed into our graph format with the following code.

```
import json

maxn = 203088
with open('./reddit_edges.json', 'r') as f:
  data = json.load(f)
  for i in range(maxn):
    print("t #", i)
    edges = data[str(i)];
    n = 0
    for (a,b) in edges:
      n = max(n, max(a, b)+1)
    for j in range(n):
      print("v", j, "0")
    for (a,b) in edges:
      print("e", a, b, "0")
```

6.2 Approximate Frequent Subgraph Mining

In this section we explore the performance of our algorithms when sampling transactions from a static dataset (e.g. in order to improve running times and memory usage).

First of all, for each of the datasets described before, we obtained the sample sizes using both the method based on the c-bound (for various maximum subgraph sizes) and the one based on the Rademacher averages. We fixed $\delta = 0.05$ (we also tested other values, but since it has only a logarithmic dependency on the sample size, the results are almost not affected by it), and let ε range in $\{0.1, 0.05, 0.02, 0.01\}$. The comparison of the sample sizes is shown in Fig. 6.1, which reports the mean values over 5 runs, together with 95% confidence interval (in shaded color). As shown in the plots, in all the datasets we tested the methods on the VC-dimension-based bounds significantly outperform the bounds based on Rademacher averages. This might be due to the progressive sampling approach employed by the Rademacher averages bound, which may lead to conservative (i.e., larger) estimates of the required sample sizes, or by the fact that Massart's lemma produces a slack bound to the Rademacher average.

Another interesting pattern that we see is that as the maximum subgraph size in the c-bound bound decreases, the sample size decreases as well. Thanks to this property, when one wants to limit the mining process to small

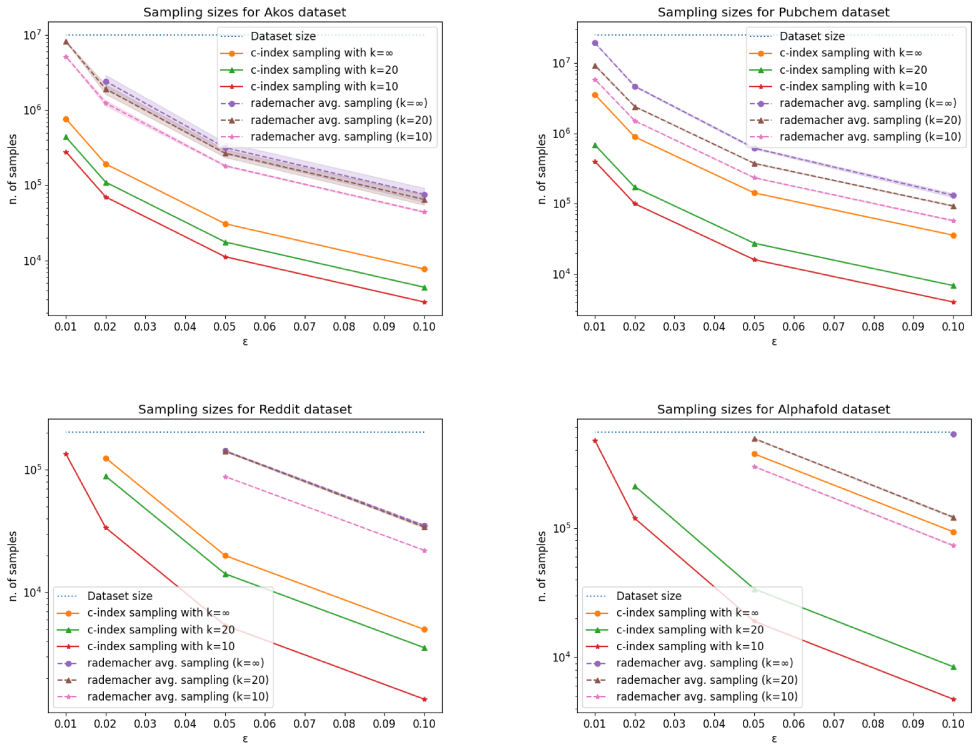

Figure 6.1: Sample size bounds using the VC dimension bound based on the c-bound and using Rademache averages. Values are mean over 5 runs. 95% confidence interval are shaded.

subgraphs, one can obtain significantly smaller sample sizes. The same holds, albiet to a more limited extent, also when limiting the maximum pattern size in the Rademacher-averages-based method.

Moreover, we compare the running times and peak memory usage of both our methods on the above datasets. We run the sampling methods 5 times and report the mean of the quantities, together with the standard deviation. As shown in Table 6.2, the c-bound-based method is faster for all datasets except Pubchem, which is the largest one. Indeed, the Rademacher-based method takes more time for each processed graph, but it also can avoid processing all the graphs of the dataset. In large datasets then this latter method then results faster. Moreover, although we omit these results for

Table 6.2: Running time and peak memory usage to obtain the sample for approximate frequent subgraph mining ($\varepsilon = 0.1$).

	c-bound		Rademacher ($\varepsilon = 0.05$)	
Dataset	Time (s)	Mem. (MB)	Time (s)	Mem. (MB)
Akos	108 ± 2	23 ± 1	113 ± 13	36 ± 1
Pubchem	270 ± 1	102 ± 0	251 ± 3	47 ± 0
Reddit	1.2 ± 0.1	21 ± 0	34.6 ± 0.8	231 ± 1
Alphafold	95 ± 2	643 ± 0	1900 ± 11	55 ± 0

Table 6.3: Precision, recall and maximum deviation of approximate solutions

		$\varepsilon = 0.01$			$\varepsilon = 0.05$		
Dataset		Prec.	Recall	Max dev.	Prec.	Recall	Max dev.
Akos	$\theta = 0.8$	0.99	1.0	0.0012	0.81	1.0	0.0046
	$\theta = 0.5$	0.95	1.0	0.0019	0.77	1.0	0.0112
	$\theta = 0.2$	0.89	1.0	0.0025	0.51	1.0	0.0112
Pubchem	$\theta = 0.8$	0.85	1.0	0.0014	0.71	1.0	0.0050
	$\theta = 0.5$	0.96	1.0	0.0015	0.78	1.0	0.0113
	$\theta = 0.2$	0.90	1.0	0.0015	0.52	1.0	0.0113
Alphafold	$\theta = 0.5$	0.99	1.0	0.007	0.93	1.0	0.0084
	$\theta = 0.2$	0.90	1.0	0.0008	0.56	1.0	0.0091
	$\theta = 0.1$	0.87	1.0	0.0008	0.47	1.0	0.0091

space constraints, we noticed that computing the c-bound becomes faster as k decreases, as the running times depend on the actual c-bound, and does not depend on ε, as the entire dataset has to be considered anyways. On the other hand, the Rademacher-based method performance grows quadratically with $1/\varepsilon$, as it depends on the the number of samples to be considered.

We then analyse the results of the frequent subgraph mining procedure on the full datasets as well as on the subsampled datasets corresponding to $\varepsilon = 0.01, 0.05$. Since the c-bound-based method has proven to be the best one for this particular task, we use its bounds, using $k = 10$ for Alphafold and $k = 20$ for the other datasets, which are conservative bounds to the maximum size of a frequent subgraph. We run the mining procedure with multiple frequency thresholds ($\theta = 0.8, 0.5, 0.2$ for all datasets except ALPHAFOLD,

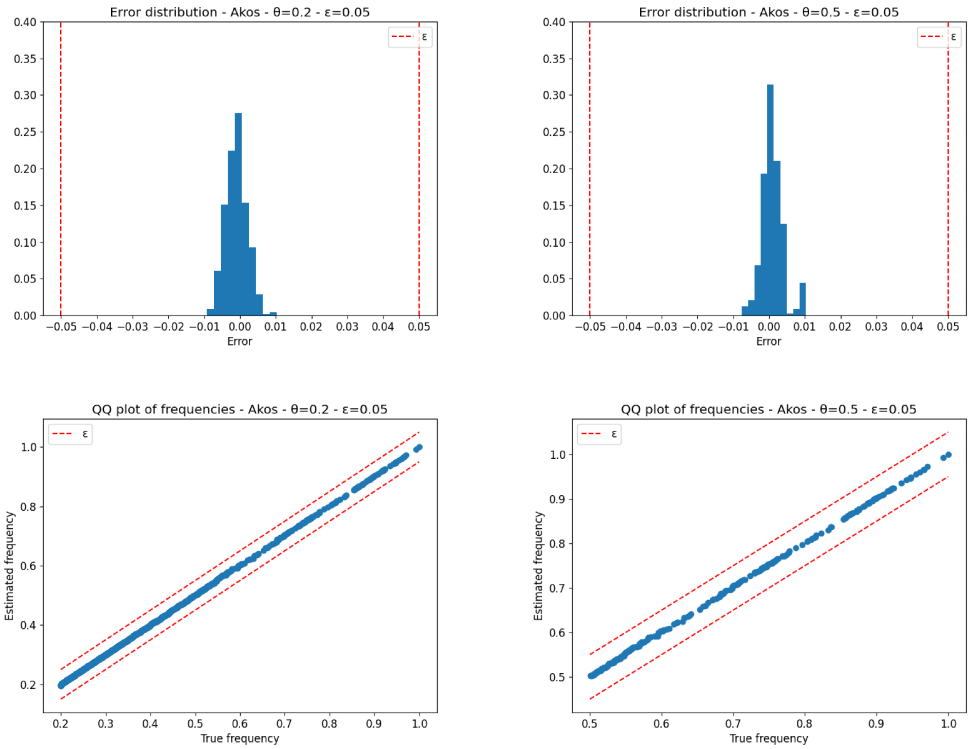

Figure 6.2: Deviation distribution of estimated frequencies for approximate subgraph mining. We use the VC dimension bound approach with $k = 10$ for Alphafold and $k = 20$ for the other datasets.

where we used $\theta = 0.5, 0.2, 0.1$ as the number of frequent patterns at higher thresholds is particularly low). Note that, as stated before, we had to use Gaston RE for the unlabelled dataset (REDDIT) as the memory usage of the standard Gaston exceeded 1Tb of RAM even for the smallest subsampled version of the dataset. Note that the output of the two versions of the miner is exactly the same. Since this latter version is quite slow, we managed to complete the mining procedure only for the subsampled dataset corresponding to $\varepsilon = 0.05$, as for larger sample sizes the computation would have exceeded several weeks of computing time. Moreover, when mining this dataset we limited the subgraph sizes to 7, as already for 8 the computation needed several weeks.

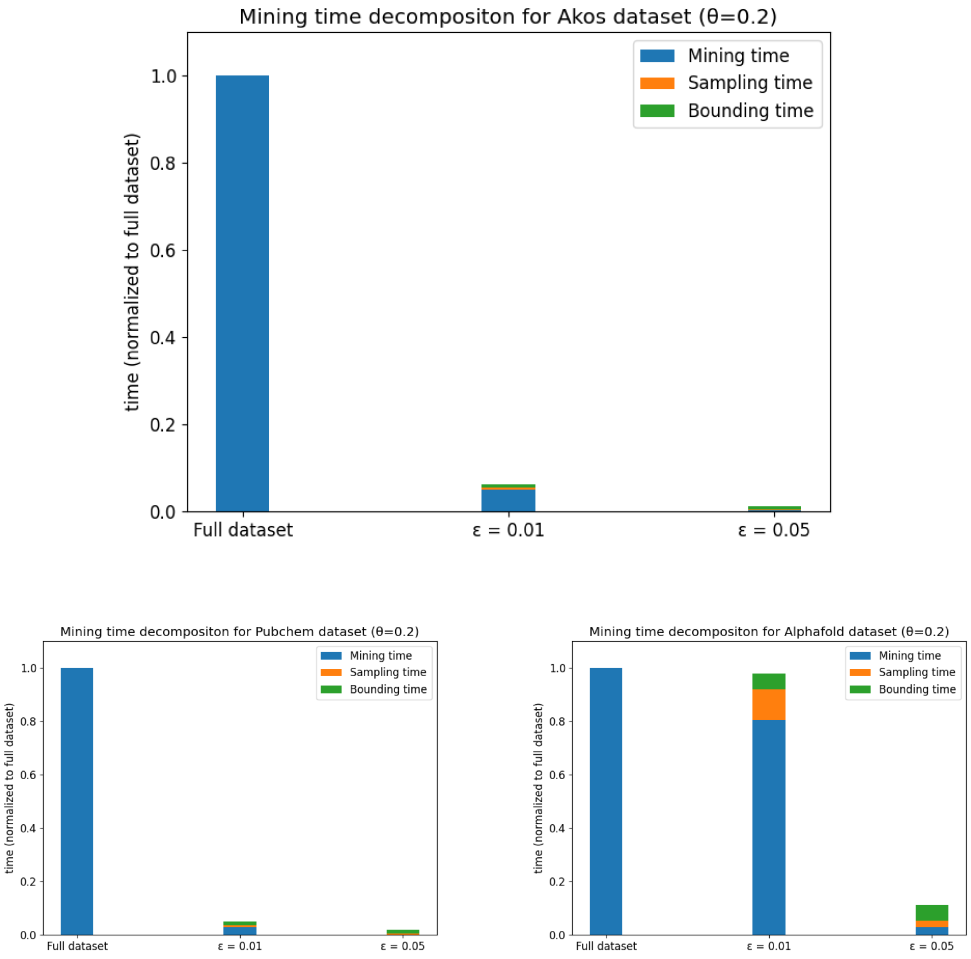

Figure 6.3: Running time decomposition for approximate frequent subgraph mining.

Table 6.4: Performance of exact and approximate mining algorithms

Dataset		Full dataset			$\varepsilon = 0.01$		
		Samples	Time (s)	Mem. (GB)	Samples	Time (s)	Mem. (GB)
Akos	$\theta = 0.8$	$10 \cdot 10^6$	690 ± 2	93 ± 0	439979	30.8 ± 0.4	4.3 ± 0
	$\theta = 0.5$		2030 ± 297	105 ± 0		82 ± 2	4.8 ± 0
	$\theta = 0.2$		9160 ± 534	118 ± 0		462 ± 45	5.5 ± 0
Pubchem	$\theta = 0.8$	$25 \cdot 10^6$	1400 ± 10	227 ± 0	684979	38 ± 0.4	6.3 ± 0
	$\theta = 0.5$		4303 ± 454	251 ± 0		105 ± 8	6.9 ± 0
	$\theta = 0.2$		14415 ± 713	279 ± 0		395 ± 25	7.7 ± 0
Reddit	$\theta = 0.8$	203088	-	-	69979	-	-
	$\theta = 0.5$		-	-		-	-
	$\theta = 0.2$		-	-		-	-
Alphafold	$\theta = 0.5$	541374	157 ± 4	9.3 ± 0	474979	144 ± 14	8.2 ± 0
	$\theta = 0.2$		310 ± 56	9.5 ± 0		302 ± 36	8.3 ± 0
	$\theta = 0.1$		542 ± 78	9.7 ± 0		436 ± 28	8.5 ± 0

Dataset		$\varepsilon = 0.05$		
		Samples	Time (s)	Mem. (GB)
Akos	$\theta = 0.8$	17599	1.1 ± 0.1	0.1 ± 0
	$\theta = 0.5$		2.0 ± 0.1	0.2 ± 0
	$\theta = 0.2$		12.5 ± 1.1	0.2 ± 0
Pubchem	$\theta = 0.8$	27400	1.4 ± 0	0.2 ± 0
	$\theta = 0.5$		3.8 ± 0.3	0.3 ± 0
	$\theta = 0.2$		13.6 ± 0.9	0.3 ± 0
Reddit	$\theta = 0.8$	2800	32867 ± 3293	0.01 ± 0
	$\theta = 0.5$		46357 ± 11170	0.01 ± 0
	$\theta = 0.2$		81780 ± 18619	0.01 ± 0
Alphafold	$\theta = 0.5$	19000	4.6 ± 0.2	0.3 ± 0
	$\theta = 0.2$		9.2 ± 1.3	0.3 ± 0
	$\theta = 0.1$		14.8 ± 3.1	0.3 ± 0

We measured (see Table 6.3) the recall, i.e. the ratio of the number of true positives and the number of actually frequent patterns, the precision, i.e. the ratio of the number of true positives and the number of returned patterns, and the maximum deviation of the reported frequency of the true positives. The reported precisions and recalls are averages over 5 runs, and since the standard deviation is always below 0.01, we omit it. The reported maximum deviations are the maxima over the 5 runs. Remarkably, the number of false negatives, throughout all of our tests, is always zero (and not only with probability $1 - \delta$), and hence our approximate mining method has always recall 1.0. For what concerns the precision, we point out that our methods have no control over it, as it depends on the number of subgraphs with frequency in $[\theta - \varepsilon, \theta]$, which is unknown a priori. Nonetheless, the experiments show that, especially for $\varepsilon = 0.01$, the number of false positives is quite limited. Moreover, all false positives are "acceptable" ones, in the sense that their frequencies are

never more than 2ε lower than the threshold, as required by Definition 3. Finally, the maximum deviation of the frequency of truly frequent patterns is noticeably smaller than the theoretical bound provided by the theory, and, as shown in Fig. 6.2, the errors are quite concentrated around 0, especially for high thresholds. This hints that tighter bounds might be possible to obtain. We remark that for computing the above quantities, we first had to solve the subgraph mining problem on the original full dataset, which is an expensive computation that in some cases (e.g. on the REDDIT dataset) is basically impossible to carry out. In these latter cases the theoretical bounds we provided are the *only guarantees* that one might have on the quality of the solution on the sample.

Lastly, we investigate the performance gains in executing the mining algorithm on the sampled datasets in terms of running times and peak memory consumption. The performance of the mining algorithm not only depends on the number of transactions in the dataset, but also on the number of frequent patterns. Since in the sampled datasets we have to mine all patterns with frequency $\theta - \varepsilon$, the number of outputted patterns will be higher than the number of true frequent patterns, and this could worsen running times and memory consumption. Note that the performance benefits are the primary reason to use the sampled datasets, hence we hope that the reduction in the input size overcomes the increase in the output size in terms of impact on the performance, and to hence see substantial improvements. Indeed, as reported in Table 6.2, we see up to a 20x reduction in peak memory usage and running times for $\varepsilon = 0.01$ and up to a 1000x reduction for $\varepsilon = 0.05$. This makes it possible to run the subgraph mining procedure on a large dataset such as PUBCHEM on a standard laptop with 8GB or RAM rather than on a dedicated system with hundreds of gigabytes of RAM. Moreover, for challenging datasets such as REDDIT, the sampling scheme allows to conclude the mining procedure in a few days of computation rather than in weeks or months, making the problem from virtually impossible to solve to solvable. We also report how the total computing time is subdivided among computing the sample size and sampling the dataset, and running the mining procedure on the sample. We remark that, especially for low values of ε, the time needed

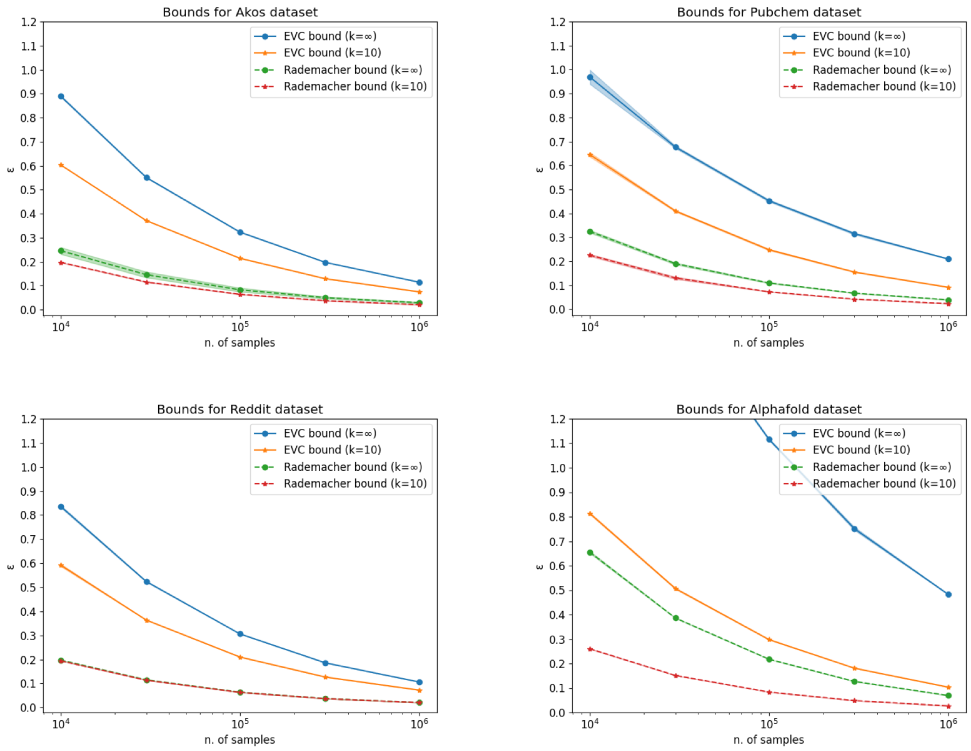

Figure 6.4: EVC vs Rademacher bounds for true frequent subgraph mining

to compute the bound and to sample the dataset is quite small compared to the time needed to actually mine the frequent patterns, as shown in Fig. 6.3.

6.3 True Frequent Subgraph Mining

As described before, another interesting formulation of the problem is when the sample is not from a static and known dataset, but rather from an unknown generating distribution. To produce the samples for our experiments we use the datasets described in Table 6.1 as the generating distribution, and sample them uniformly and with replacement. We generated, for each dataset, a set of samples of sizes in the range $[10^4, 10^6]$.

As stated in Section 5.2, since in general we don't have access to the generating distribution, we have to use the bounds based on the empirical

VC dimension rather than the full VC dimension, since the latter is unknown, and this leads to much weaker bounds. The bounds based on Rademacher averages instead are not based on information on the distribution, but rather on the samples only, and hence are still valid. As shown in Fig. 6.4, in this framework the Rademacher-complexity-based bounds significantly outperform the empirical-VC-based ones. Moreover, limiting the maximum pattern size helps produce sharper bounds. Once again, the reported quantities are averages over 5 runs together with 95% confidence intervals.

Finally, we exploited the bounds derived from the Rademacher-averages-based methods, since they are the best ones, to mine the samples ad a lowered threshold and obtain an approximation to the true frequent itemsets in the generating distribution. Since the distributions reflects the datasets of Table 6.1, we have access to the probability distribution and can thus calculate the precision, recall and maximum deviation of the approximate frequencies, which are reported in Table 6.5. As expected, as the bound on the maximum deviation decreases with the increasing sample size, the precision of the appriximate mining algorithm increases for all datasets and thresholds. Moreover, as in the other variant of the problem, the recall of the method is always 1.0, that is there are no false neagatives. It is interesting to notice that the actual maximum deviations of the estimated frequencies from the true ones are uch smaller that the bounds provided by Rademacher averages. This suggests that sharper bounds might be found.

Table 6.5: Precision, recall and maximum deviation when sampling from a distribution

		$n = 10^4$				$n = 10^5$			
Dataset		Bound	Precis.	Recall	Max dev.	Bound	Precis.	Recall	Max dev.
Akos	$\theta = 0.8$	0.232	0.39	1.0	0.0064	0.075	0.62	1.0	0.0018
	$\theta = 0.5$		0.15	1.0	0.0124		0.62	1.0	0.0038
	$\theta = 0.2$		-	-	-		0.32	1.0	0.0040
Pubchem	$\theta = 0.8$	0.285	0.24	1.0	0.0051	0.091	0.58	1.0	0.0032
	$\theta = 0.5$		0.13	1.0	0.0165		0.69	1.0	0.0043
	$\theta = 0.2$		-	-	-		0.26	1.0	0.0053
Alphafold	$\theta = 0.5$	0.260	0.48	1.0	0.0077	0.083	0.91	1.0	0.0041
	$\theta = 0.2$		-	-	-		0.36	1.0	0.0041
	$\theta = 0.1$		-	-	-		-	-	-

		$n = 10^6$			
Dataset		Bound	Precis.	Recall	Max dev.
Akos	$\theta = 0.8$	0.026	0.93	1.0	0.0008
	$\theta = 0.5$		0.84	1.0	0.0015
	$\theta = 0.2$		0.72	1.0	0.0015
Pubchem	$\theta = 0.8$	0.029	0.79	1.0	0.0012
	$\theta = 0.5$		0.89	1.0	0.0014
	$\theta = 0.2$		0.70	1.0	0.0016
Alphafold	$\theta = 0.5$	0.026	0.97	1.0	0.0010
	$\theta = 0.2$		0.74	1.0	0.0017
	$\theta = 0.1$		0.19	1.0	0.0017

Chapter 7

Conclusions

In this thesis, we derived efficiently computable bounds on the VC-dimension and on the Rademacher averages of subgraphs. We showed that bounds can be used to obtain efficient approximations for two graph mining tasks: frequent subgraph mining and true frequent subgraph mining. Our extensive experimental evaluations shows that our bounds, and the corresponding algorithms, result in high-quality approximations for both applications on several real datasets.

Bibliography

[AIS93] Rakesh Agrawal, Tomasz Imielinski, and Arun N. Swami. Mining association rules between sets of items in large databases. In Peter Buneman and Sushil Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, DC, USA, May 26-28, 1993*, pages 207–216. ACM Press, 1993.

[ALA19] Isam A Alobaidi, Jennifer Leopold, and Ali A Allami. The use of frequent subgraph mining to develop a recommender system for playing real-time strategy games. In *Proceedings of the 2019 IEEE International Conference on Data Mining (ICDM 2019)*, pages 146–160, 2019.

[AS94] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In Jorge B. Bocca, Matthias Jarke, and Carlo Zaniolo, editors, *Proceedings of 20th International Conference on Very Large Data Bases (VLDB 1994), September 12-15, 1994, Santiago de Chile, Chile*, pages 487–499, 1994.

[BB02] Christian Borgelt and Michael R. Berthold. Mining molecular fragments: Finding relevant substructures of molecules. In *Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM 2002), 9-12 December 2002, Maebashi City, Japan*, pages 51–58. IEEE Computer Society, 2002.

[BBL05] Stéphane Boucheron, Olivier Bousquet, and Gábor Lugosi. Theory of classification: a survey of some recent advances. *ESAIM: Probability and Statistics*, 9:323–375, 2005.

[BEHW89] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Learnability and the vapnik-chervonenkis dimension. *J. ACM*, 36(4):929–965, 1989.

[CB11] B. Chandra and Shalini Bhaskar. A new approach for generating efficient sample from market basket data. *Expert Syst. Appl.*, 38(3):1321–1325, 2011.

[CFSV18] Vincenzo Carletti, Pasquale Foggia, Alessia Saggese, and Mario Vento. Challenging the time complexity of exact subgraph isomorphism for huge and dense graphs with vf3. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):804–818, 2018.

[CHGM01] Diane J. Cook, Lawrence B. Holder, Gehad Galal, and Ron Maglothin. Approaches to parallel graph-based knowledge discovery. *J. Parallel Distributed Comput.*, 61(3):427–446, 2001.

[CPS09] Venkatesan T Chakaravarthy, Vinayaka Pandit, and Yogish Sabharwal. Analysis of sampling techniques for association rule mining. In *Proceedings of the 12th International Conference on Database Theory (ICDT 2009)*, pages 276–283, 2009.

[DKWK05] Mukund Deshpande, Michihiro Kuramochi, Nikil Wale, and George Karypis. Frequent substructure-based approaches for classifying chemical compounds. *IEEE Transactions on Knowledge and Data Engineering*, 17(8):1036–1050, 2005.

[EASK14] Mohammed Elseidy, Ehab Abdelhamid, Spiros Skiadopoulos, and Panos Kalnis. Grami: Frequent subgraph and pattern mining in a

single large graph. *Proceedings of the VLDB Endowment*, 7(7):517–528, 2014.

[FL11] Dan Feldman and Michael Langberg. A unified framework for approximating and clustering data. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 569–578, 2011.

[FSS20] Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: Constant-size coresets for k-means, pca, and projective clustering. *SIAM J. Comput.*, 49(3):601–657, 2020.

[FWWX15] Wenfei Fan, Xin Wang, Yinghui Wu, and Jingbo Xu. Association rules with graph patterns. *Proceedings of the VLDB Endowment*, 8(12):1502–1513, 2015.

[GC02] Shayan Ghazizadeh and Sudarshan S. Chawathe. Seus: Structure extraction using summaries. In *Proceedings of the 2002 International Conference on Discovery Science*, pages 71–85. Springer, 2002.

[GK01] Valerie Guralnik and George Karypis. A scalable algorithm for clustering sequential data. In *Proceedings of the 2001 IEEE International Conference on Data Mining (ICDM 2001)*, pages 179–186. IEEE, 2001.

[HPYM04] Jiawei Han, Jian Pei, Yiwen Yin, and Runying Mao. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Min. Knowl. Discov.*, 8(1):53–87, 2004.

[HS11] Sariel Har-Peled and Micha Sharir. Relative (p, ε)-approximations in geometry. *Discret. Comput. Geom.*, 45(3):462–496, 2011.

[HSS12] Steven Hill, Bismita Srichandan, and Rajshekhar Sunderraman. An iterative mapreduce approach to frequent subgraph mining in biological datasets. In Sanjay Ranka, Tamer Kahveci, and Mona Singh, editors, *ACM International Conference on Bioinformatics, Computational Biology and Biomedicine, BCB' 12, Orlando, FL, USA - October 08 - 10, 2012*, pages 661–666. ACM, 2012.

[HWP03] Jun Huan, Wei Wang, and Jan F. Prins. Efficient mining of frequent subgraphs in the presence of isomorphism. In *Proceedings of the*

3rd IEEE International Conference on Data Mining (ICDM 2003), 19-22 December 2003, Melbourne, Florida, USA, pages 549–552. IEEE Computer Society, 2003.

[IWM00] Akihiro Inokuchi, Takashi Washio, and Hiroshi Motoda. An apriori-based algorithm for mining frequent substructures from graph data. In *Proceedings of the 2000 European Conference on Principles of Data Mining and Knowledge Discovery*, pages 13–23. Springer, 2000.

[JEP+21] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, July 2021.

[JL96] George H. John and Pat Langley. Static versus dynamic sampling for data mining. In Evangelos Simoudis, Jiawei Han, and Usama M. Fayyad, editors, *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA*, pages 367–370. AAAI Press, 1996.

[KK01] Michihiro Kuramochi and George Karypis. Frequent subgraph discovery. In Nick Cercone, Tsau Young Lin, and Xindong Wu, editors, *Proceedings of the 2001 IEEE International Conference on Data Mining (ICDM 2001), 29 November - 2 December 2001, San Jose, California, USA*, pages 313–320. IEEE Computer Society, 2001.

[KP00] Vladimir Koltchinskii and Dmitriy Panchenko. Rademacher processes and bounding the risk of function learning. In *High dimensional probability II*, pages 443–457. Springer, 2000.

[LMR91] Nathan Linial, Yishay Mansour, and Ronald L Rivest. Results on learnability and the vapnik-chervonenkis dimension. *Information and Computation*, 90(1):33–49, 1991.

[LP09] Maarten Löffler and Jeff M. Phillips. Shape fitting on point sets with probability distributions. In Amos Fiat and Peter Sanders, editors, *Proceedings of the 17th Annual European Symposium on Algorithms (ESA 2009), Copenhagen, Denmark, September 7-9, 2009.*, volume 5757 of *Lecture Notes in Computer Science*, pages 313–324. Springer, 2009.

[LXG14] Wenqing Lin, Xiaokui Xiao, and Gabriel Ghinita. Large-scale frequent subgraph mining in mapreduce. In Isabel F. Cruz, Elena Ferrari, Yufei Tao, Elisa Bertino, and Goce Trajcevski, editors, *IEEE 30th International Conference on Data Engineering (ICDE 2014), Chicago, IL, USA, March 31 - April 4, 2014*, pages 844–855. IEEE Computer Society, 2014.

[McD89] Colin McDiarmid. *On the method of bounded differences*, pages 148–188. London Mathematical Society Lecture Note Series. Cambridge University Press, 1989.

[MMB+18] Aida Mrzic, Pieter Meysman, Wout Bittremieux, Pieter Moris, Boris Cule, Bart Goethals, and Kris Laukens. Grasping frequent subgraph mining for bioinformatics applications. *BioData mining*, 11(1):1–24, 2018.

[MTV94] Heikki Mannila, Hannu Toivonen, and A. Inkeri Verkamo. Efficient algorithms for discovering association rules. In Usama M. Fayyad and Ramasamy Uthurusamy, editors, *Knowledge Discovery in Databases: Papers from the 1994 AAAI Workshop, Seattle, Washington, USA, July 1994. Technical Report WS-94-03*, pages 181–192. AAAI Press, 1994.

[NK05] Siegfried Nijssen and Joost N. Kok. The gaston tool for frequent subgraph mining. *Electron. Notes Theor. Comput. Sci.*, 127(1):77–87, 2005.

[NLo] Nlopt. https://nlopt.readthedocs.io/.

[PCVR20] Leonardo Pellegrina, Cyrus Cousins, Fabio Vandin, and Matteo Riondato. Mcrapper: Monte-carlo rademacher averages for poset families and approximate pattern mining. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2165–2174, 2020.

[PDFMR21] Giulia Preti, Gianmarco De Francisci Morales, and Matteo Riondato. Maniacs: Approximate mining of frequent subgraph patterns through sampling. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1348–1358, 2021.

[Pol12] David Pollard. *Convergence of stochastic processes*. Springer Science & Business Media, 2012.

[RAÇ+11] Matteo Riondato, Mert Akdere, Uğur Çetintemel, Stanley B Zdonik, and Eli Upfal. The vc-dimension of sql queries and selectivity estimation through sampling. In *Proceedings of the 2011 Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 661–676. Springer, 2011.

[RU14] Matteo Riondato and Eli Upfal. Efficient discovery of association rules and frequent itemsets through sampling with tight performance guarantees. *ACM Trans. Knowl. Discov. Data*, 8(4), aug 2014.

[RU15] Matteo Riondato and Eli Upfal. Mining frequent itemsets through progressive sampling with rademacher averages. In Longbing Cao, Chengqi Zhang, Thorsten Joachims, Geoffrey I. Webb, Dragos D. Margineantu, and Graham Williams, editors, *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, pages 1005–1014. ACM, 2015.

[RU18] Matteo Riondato and Eli Upfal. ABRA: approximating betweenness centrality in static and dynamic graphs with rademacher averages. *ACM Trans. Knowl. Discov. Data*, 12(5):61:1–61:38, 2018.

[RV14] Matteo Riondato and Fabio Vandin. Finding the true frequent itemsets. In Mohammed Javeed Zaki, Zoran Obradovic, Pang-Ning Tan, Arindam Banerjee, Chandrika Kamath, and Srinivasan Parthasarathy, editors, *Proceedings of the 2014 SIAM International Conference on Data Mining, Philadelphia, Pennsylvania, USA, April 24-26, 2014*, pages 497–505. SIAM, 2014.

[RV20] Matteo Riondato and Fabio Vandin. Misosoup: Mining interesting subgroups with sampling and pseudodimension. *ACM Trans. Knowl. Discov. Data*, 14(5):56:1–56:31, 2020.

[SSBD14] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

[SSRZ20] Sacha Servan-Schreiber, Matteo Riondato, and Emanuel Zgraggen. Prosecco: Progressive sequence mining with convergence guarantees. *Knowledge and Information Systems*, 62(4):1313–1340, 2020.

[STV20] Diego Santoro, Andrea Tonon, and Fabio Vandin. Mining sequential patterns with vc-dimension and rademacher complexity. *Algorithms*, 13(5):123, 2020.

[Vap98] Vladimir N Vapnick. *Statistical learning theory*. Wiley, New York, 1998.

[VC82] V. N. Vapnik and A. Ya. Chervonenkis. Necessary and sufficient conditions for the uniform convergence of means to their expectations. *Theory of Probability & Its Applications*, 26(3):532–553, 1982.

[VC13] Vladimir Naumovich Vapnik and Alexey Ya. Chervonenkis. On the uniform convergence of the frequencies of occurrence of events to their probabilities. In Bernhard Schölkopf, Zhiyuan Luo, and Vladimir Vovk, editors, *Empirical Inference - Festschrift in Honor of Vladimir N. Vapnik*, pages 7–12. Springer, 2013.

[WMFP05] Marc Wörlein, Thorsten Meinl, Ingrid Fischer, and Michael Philippsen. A quantitative comparison of the subgraph miners mofa, gspan, ffsm, and gaston. In Alípio Mário Jorge, Luís Torgo, Pavel Brazdil, Rui

Camacho, and João Gama, editors, *Knowledge Discovery in Databases: PKDD 2005*, pages 392–403, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[WWP+04] Chen Wang, Wei Wang, Jian Pei, Yongtai Zhu, and Baile Shi. Scalable mining of large disk-based graph databases. In Won Kim, Ron Kohavi, Johannes Gehrke, and William DuMouchel, editors, *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22-25, 2004*, pages 316–325. ACM, 2004.

[YH02] Xifeng Yan and Jiawei Han. gspan: Graph-based substructure pattern mining. In *Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM 2002), 9-12 December 2002, Maebashi City, Japan*, pages 721–724. IEEE Computer Society, 2002.