A PAC-Theory of Clustering with Advice

by

Mohammad Zokaei Ashtiani

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science

Waterloo, Ontario, Canada, 2018

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the examining Committee is by majority vote.

External Examiner:	Maria-Florina Balcan Associate Professor, Dept. of Computer Science, Carnegie Mellon University
Supervisor(s):	Shai Ben-David Professor, Dept. of Computer Science, University of Waterloo
Internal Member:	Pascal Poupart Professor, Dept. of Computer Science, University of Waterloo
Internal Member:	Yaoliang Yu Assistant Professor, Dept. of Computer Science, University of Waterloo
Internal-External Member:	Ali Ghodsi Professor, Dept. of Statistics and Actuarial Science, University of Waterloo

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

Parts of this dissertation are based on some publications that I have co-authored. In particular, Chapter 3 is based on a joint work with Shai Ben-David [9]. Also, Chapter 4 is based on a joint work with Shrinu Kushagra and Shai Ben-David [14]. Chapters 6 and 7 are based on joint works with Abbas Mehrabian and Shai Ben-David [12, 11]. Finally, a much more complete version of [11] was later prepared together with Shai Ben-David, Nick Harvey, Chris Liaw, Abbas Mehrabian and Yaniv Plan [10].

Abstract

In the absence of domain knowledge, clustering is usually an under-specified task. For any clustering application, one can choose among a variety of different clustering algorithms, along with different preprocessing techniques, that are likely to result in dramatically different answers. Any of these solutions, however, can be acceptable depending on the application, and therefore, it is critical to incorporate prior knowledge about the data and the intended semantics of clustering into the process of clustering model selection.

One scenario that we study is when the user (i.e., the domain expert) provides a clustering of a (relatively small) random subset of the data set. The clustering algorithm then uses this kind of "advice" to come up with a data representation under which an application of a fixed clustering algorithm (e.g., k-means) results in a partition of the full data set that is aligned with the user's knowledge. We provide "advice complexity" of learning a representation in this paradigm.

Another form of "advice" can be obtained by allowing the clustering algorithm to interact with a domain expert by asking *same-cluster queries*: "Do these two instances belong to the same cluster?". The goal of the clustering algorithm will then be finding a partition of the data set that is consistent with the domain expert's knowledge (yet using only a small number of queries). Aside from studying the "advice complexity" (i.e., query complexity) of learning in this model, we investigate the trade-offs between computational and advice complexities of learning, showing that using a little bit of advice can turn an otherwise computationally hard clustering problem into a tractable one.

In the second part of this dissertation we study the problem of learning mixture models, where we are given an i.i.d. sample generated from an unknown target from a family of mixture distributions, and want to output a distribution that is close to the target in total variation distance. In particular, given a sample-efficient learner for a base class of distributions (e.g., Gaussians), we show how one can come up with a sample-efficient method for learning mixtures of the base class (e.g., mixtures of k Gaussians). As a byproduct of this analysis, we are able to prove tighter sample complexity bounds for learning various mixture models. We also investigate how having access to the same-cluster queries (i.e., whether two instances were generated from the same mixture component) can help reducing the computational burden of learning within this model.

Finally, we take a further step and introduce a novel method for distribution learning via a form of *compression*. In particular, we ask whether one can compress a large-enough sample set generated from a target distribution (by picking only a few instances from it) in a way that allows recovery of (an approximation to) the target distribution. We prove that if this is the case for all members of a class of distributions, then there is a sample-efficient way of distribution learning with respect to this class. As an application of this novel notion, we settle the sample complexity of learning mixtures of k axis-aligned Gaussian distributions (within logarithmic factors).

Acknowledgements

My sincere gratitude goes to my supervisor, Professor Shai Ben-David, who helped me patiently throughout this long journey. I was extremely lucky to have such a brilliant advisor and teacher. I would not have been able to deliver this dissertation without his support.

I would like to thank my thesis committee members – Professor Ali Ghodsi, Professor Pascal Poupart, Professor Yaoliang Yu and Professor Maria-Florina Balcan – for providing valuable feedback and comments about my work.

I would like to thank Abbas Mehrabian and Shrinu Kushagra who are also the co-authors of parts of this dissertation. I am also grateful to Vinayak Pathak and Samira Samadi for the helpful discussions regarding the topics of this dissertation.

I will not forget my marvelous time in Waterloo with my incredible friends.

My deepest appreciation goes to my lovely parents, Sima and Morteza, for basically everything I have accomplished. I am thankful to my brother, Mojtaba, for his constant support, and to my sister, Mahya, for her sincere encouragements.

Most importantly, I would like to thank Elnaz, my best friend and my beloved wife. Without her help and support, I would have simply given up. She made my graduate life joyful, cheered me up in my weak moments, and literally helped me with my research. I feel extremely blessed to have her by my side.

Dedication

In loving memory of Parisa.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Clustering can be thought as the task of automatically dividing a set of objects into "coherent" subsets. This definition is not concrete, but its vagueness allows it to serve as an umbrella term for a wide diversity of algorithmic paradigms. Clustering algorithms are being routinely applied in a huge variety of fields.

Clustering is a challenging task particularly due to two impediments. The first problem is that clustering, in the absence of domain knowledge, is usually an *under-specified* task; the solution of choice may vary significantly between different intended applications. The second one is that performing clustering under many natural models is computationally hard.

Consider the task of dividing the users of an online shopping service into different groups. The result of this clustering can then be used for example in suggesting similar products to the users in the same group, or for organizing data so that it would be easier to read/analyze the monthly purchase reports. Those different applications may result in conflicting solution requirements. In such cases, one needs to exploit domain knowledge to better define the clustering problem.

Technically speaking, given a dataset that needs to be clustered for some application, one can choose among a variety of different clustering algorithms, along with different preprocessing techniques, that are likely to result in dramatically different answers. Any of these solutions, however, can be acceptable, and therefore it is critical to incorporate prior knowledge about the data and the intended clustering semantics into the process of *model selection* for clustering.

Regretfully, many of the existing approaches for incorporation of domain knowledge into clustering—which are sometimes called semi-supervised clustering methods—are not systematic, and involve trial-and-error or follow embarrassingly *ad hoc* measures. A major goal of this dissertation is to address this shortcoming.

In particular, we would like to propose and study multiple notions of “advice” that can make the clustering problem well-defined. Note that *advice* is a generic term that we use to call the various types of “hints” about the clustering problem which are provided by a domain expert. For example, advice can be in the form of some constraints (on the final clustering solution) that are given off-line, or be the answers to some queries that are adaptively asked by the clustering algorithm.

Furthermore, we want to define and analyze the “advice complexity” of clustering problems in a formal framework. In other words, we would like to know *how much advice* is enough to guarantee finding an (approximately) optimal solution.

Aside from the information-theoretic aspects, we plan to study the *computational* benefits of *advice* as well. More interestingly, we would like to see if using a little bit of advice from an expert (or an oracle) can turn a computationally hard clustering problem into a tractable one. Proving such results is the second major theme in this dissertation.

Finally, we would like to take a further step and investigate the usability of *advice* in other unsupervised learning problems, such as *density estimation* and *learning mixture models*. In particular, we are interested to investigate the effect of *advice* on the computational and statistical complexities of those problems.

Our aim is to address the mentioned problems in a formal framework. Therefore, we will rely on mathematical proofs rather than simulations and experiments. In particular, we will often require finding a *probably approximately correct (PAC)* solution for learning problems. The specific settings and the details of each framework differ from problem to problem, and much of our effort has been devoted to developing novel formal frameworks that (i) make sense in practice and (ii) can be rigorously analyzed.

1.1 Objectives

In the previous section we alluded to the high level goals of this dissertation. In this section we make our objectives more concrete.

1.1.1 Formalization of Clustering with Advice

The starting point of our research was seeking new ways of incorporating domain knowledge into clustering within a formal framework. Accordingly, developing new learning models is an essential goal of this work, which can be made possible by answering the following questions.

Communication Protocol. How should the learner and the domain expert communicate? What kind of (off-line or interactive) protocol can we develop that is both *user-friendly* and *effective*?

Performance Measure. What *objective function* should we use to evaluate the performance of clustering with advice?

Model. What kinds of models can we use to *encode* domain expert's knowledge? For instance, expert's intuitions may be modeled as a "representation" of data or as a similarity metric between points. The class of models that we use should be rich enough to capture expert's knowledge; yet, there should be some *inductive bias* that makes solving the problem statistically possible.

Theoretical Guarantees. What types of *statistical* and *computational* guarantees should we expect from these algorithms?

Assumptions. Are there assumptions (about the data or the given advice) that can make clustering with advice possible? We ideally want realistic assumptions that hold for real world applications—those which are not oversimplifying the question yet making the problem practically and theoretically feasible.

1.1.2 Algorithms for Clustering with Advice

Our next goal is to come up with efficient solutions to the clustering problems that we formalize. More specifically, our aim is to answer these questions.

Algorithms. What kind of methods/algorithms can we use to *train* the (parameters of the) proposed models?

Advice Complexity. What is the *advice complexity* of the proposed method?

Computational Complexity What is the *computational complexity* of the proposed method?

1.1.3 Lower bounds for Clustering with Advice

We would also like to provide computational and information-theoretic limits for *any* method that one may use for clustering with advice.

Lower Bounds for Advice Complexity. What are the lower bounds that we can prove for *advice* complexity of the problem?

Lower Bounds for Computational Complexity. What are the lower bounds that we can prove for *computational* complexity of the problem?

Trade-offs. Is there a trade-off between computational complexity and advice complexity?

1.1.4 Unsupervised Learning with Advice

We are curious to see the effect of using advice not only in clustering, but also in other unsupervised learning problems. One of the applications that can benefit from such forms of advice is learning mixture models.

Learning Mixture Models. Can advice reduce the statistical or computational complexity of learning mixture models?

Other Applications. Can the tools that we develop for learning with advice make statistical or computational analysis of other problems simpler?

1.1.5 Learning Mixture Models

We mentioned that a side-goal of this dissertation is to investigate whether advice can reduce the statistical or computational complexity of learning mixture models. While studying this problem, we realized that even in the standard density estimation setting (i.e., without queries), the sample complexity of learning mixture models is an open problem. Therefore, a major technical goal of our work is to develop new techniques and sharp bounds on the sample complexity of learning mixture models.

1.2 Summary of Contributions

In each of the following subsections, we present the outline of our contributions within a specific chapter of this dissertation.

1.2.1 Representation Learning for Clustering with Advice

We address the problem of communicating domain knowledge from a user to the clustering algorithm. We propose a protocol in which the user provides a clustering of a relatively small random sample of a data set. The clustering algorithm then uses that sample to come up with a data representation under which k-means clustering results in a clustering (of the full data set) that is aligned with the user's clustering. We provide a formal statistical model for analyzing the sample complexity (i.e., advice complexity) of learning a clustering representation within this paradigm. We then introduce a notion of capacity of a class of possible representations, in the spirit of the VC-dimension, showing that classes of representations that have finite such dimension can be successfully learned with sample size error bounds. In particular, we show that for classes of representations induced by linear embeddings, this dimension grows bi-linearly with the Euclidean dimension of the source and the target spaces.

1.2.2 Efficient Clustering with Advice

We propose a framework for Semi-Supervised Active Clustering framework (SSAC), where the learner is allowed to interact with a domain expert, asking whether two given instances belong to the same cluster or not. We study the query and computational complexity of clustering in this framework. We consider a setting where the expert conforms to a center-based clustering with a notion of margin, and show that there is a trade off between computational complexity and query complexity; we prove that for the case of k-means clustering (i.e., when the expert conforms to a solution of k-means), having access to relatively few such queries allows efficient solutions to otherwise NP hard problems.

In particular, we provide a probabilistic polynomial-time (BPP) algorithm for clustering in this setting that asks $O\big(k^2 \log k + k \log n\big)$ same-cluster queries and runs with time complexity $O\big(kn \log n\big)$ (where k is the number of clusters and n is the number of instances). The algorithm succeeds with high probability for data satisfying margin conditions under which, without queries, we show that the problem is NP hard. We also prove a lower bound on the number of queries needed to have a computationally efficient clustering algorithm in this setting.

1.2.3 Learning Mixture Models with/without Advice

We consider PAC learning of probability distributions (a.k.a. density estimation), where we are given an i.i.d. sample generated from an unknown target distribution, and want to output a distribution that is close to the target in total variation distance. Let $\mathcal{F}$ be an arbitrary class of probability distributions, and let k-mix$(\mathcal{F})$ denote the class of k-mixtures of elements of $\mathcal{F}$. Assuming the existence of a method for learning $\mathcal{F}$ with sample complexity $m_{\mathcal{F}}(\varepsilon)$, we provide a method for learning k-mix$(\mathcal{F})$ with sample complexity $O(k \log k \cdot m_{\mathcal{F}}(\varepsilon)/\varepsilon^2)$.

This general result enables us to improve the best known sample complexity upper bounds for a variety of important mixture classes. First, we show that the class of mixtures of k axis-aligned Gaussians in $\mathbb{R}^d$ is PAC-learnable with $\widetilde{O}(kd/\varepsilon^4)$ samples, which is tight in k and d up to logarithmic factors. Second, we show that the class of mixtures of k Gaussians in $\mathbb{R}^d$ is PAC-learnable with sample complexity $\widetilde{O}(kd^2/\varepsilon^4)$, which improves the previous known bounds of $\widetilde{O}(k^3d^2/\varepsilon^4)$ and $\widetilde{O}(k^4d^4/\varepsilon^2)$ in its dependence on k and d. Finally, we show that the class of mixtures of k log-concave distributions over $\mathbb{R}^d$ is PAC-learnable using $\widetilde{O}(d^{(d+5)/2}\varepsilon^{-(d+9)/2}k)$ samples.

We also show how these results are related to clustering with advice. In particular, we show that using advice we can have a computationally efficient algorithm for learning mixtures, provided that an efficient algorithm for learning the base class exists.

1.2.4 Learning Mixture Models via Compression

We study sample-efficient distribution learning, where – just like the previous subsection – a learner is given an i.i.d. sample from an unknown target distribution, and aims to approximate that distribution.

We introduce a novel method for distribution learning via a form of *compression*. Having a large-enough sample from a target distribution, can one compress that sample set – by picking only a few instances from it – in a way that allows recovery of (an approximation to) the target distribution from the compressed set? We prove that if this is the case for all members of a class of distributions, then there is a sample-efficient way of distribution learning for this class.

As an application of our approach, we provide a sample-efficient method for distribution learning with respect to the class of mixtures of k axis-aligned Gaussian distributions over $\mathbb{R}^d$. This method uses only $\widetilde{O}(kd/\epsilon^2)$ samples (to guarantee with high probability an error of at most ϵ). This is the first sample complexity upper bound that is tight in k, d, and ϵ up to logarithmic factors.

Along the way, we prove several properties of compression schemes. Namely, we prove that if there is a compression scheme for a base class of distributions, then there is a compression scheme for the class of mixtures as well as the products of that base class. These closure properties make compression schemes a powerful tool. For example, the problem of learning mixtures of axis-aligned Gaussians reduces to that of compressing one-dimensional Gaussian distributions, which we show is possible using a compressed set of constant size.

1.3 How to Read this Dissertation

The dissertation is composed of two major parts. The first one is about clustering and includes Chapters 2, 3 and 4. Note that Chapters 3 and 4 are orthogonal, and it is possible to skip the first one (which is based on an older result) without having a problem grasping the other.

The second part of the thesis is focused on learning mixture models and includes Chapters 5, 6 and 7. In this case, it is recommended to look at Chapter 6 before reading Chapter 7. In a sense, Chapter 7 is an improved version of Chapter 6 for the special case of Gaussian distributions.

These two major parts of the dissertation are connected based on the fact that mixture models can be leaned with advice as well (just like clustering methods). These two parts are, however, independent and the reader can start with reading each of them that she/he is interested. Furthermore, in order to make the dissertation easier to read, we have tried to make each chapter as self-contained as possible. Moreover, some of the proofs are omitted from the main text, and postponed to the appendices at the end of the *corresponding* chapters.

Chapter 2

Clustering with Advice: Background

In this chapter we review the relevant literature on the problem of clustering with advice. Advice can be thought as a form of supervision that can help doing model selection for clustering. More generally, advice can be thought as any form of hint that the domain expert can provide about a set of instances that are going to be clustered.

Clustering with advice can be conceptually categorized as a special case of “semi-supervised learning”. However, to make it more specific to clustering (rather than to e.g., classification), some authors use the term “semi-supervised clustering” [22, 24, 59].

In the next section, we categorize semi-supervised clustering models in terms of the protocol used to convey supervision. Then, we will review different approaches to semi-supervised clustering.

2.1 Advice Protocol

The most common method to convey supervision is through a set of pairwise *must/cannot-link* constraints over the instances [86]. These constraints are sometimes called “side-information” [87]. In this setting it is usually assumed that the given data points lie in some metric space and the learner has access to the pairwise distances; however, this rough distance information is not enough for clustering and the supervised constraints should also be taken into account.

In some other scenarios, the supervised feedback is in the form of pairwise similarities [58, 47]. In this case, the goal is to learn a good clustering without seeing/measuring all the pairwise similarities. In order to reduce the amount of required supervision, usually an active setting is used where the pairwise similarities are asked by the learner gradually [58, 47]. In a related setting, [84] considered an active framework where the learner, instead of asking about a pairwise similarity, makes a one-vs-all query (which means that the similarity of the instance with all of the other instances is requested).

Inspired by the query models in concept learning [6], Balcan et al. [18] proposed an interactive setup where in each step the learner outputs a clustering, and the teacher corrects him. This correction is either in the form of a *split* advice, or a *merge* advice. This type of supervision has the advantage of being more intuitive for the domain expert (i.e., the teacher). However, for most of the large data sets with large number of clusters it is hard for the teacher to check the output of the learner in each step (unless e.g., the outcome of the clustering is meaningfully visualizable).

Another possibility is to ask the expert to provide a clustering of a small subset of instances. The learner then learns how to cluster the whole data set based on this demonstration. We will study this new setting in Chapter 3.

Yet another idea is to allow the learner to ask queries like "Do instances x_1 and x_2 belong to the same cluster?" We will study this new form of supervision in Chapter 4.

2.2 Semi-Supervised Clustering Methods

2.2.1 Constrained Clustering

Semi-supervised clustering with pairwise constraints is probably the oldest method to inject supervision into clustering. The common way of using such supervision is by changing the objective of clustering so that violation of these constraints is penalized [37, 61, 25]. These methods are sometimes called "constrained clustering".

There have been several attempts to benefit from supervision for k-means clustering. Wagstaff et al. [86] modified the well known Lloyd's algorithm [65] to avoid assigning conflicting instances to the same cluster. Also, Basu et al. [22] used labeled data to initialize the centers for the Lloyd's algorithm.

Hierarchical (i.e., agglomerative) clustering methods have also been extended to the supervised setting. In [74], pairwise constraints were used to prune the clustering tree. Davidson and Ravi [36] also studied this setting, and showed some computational hardness results about the satisfiability of these constraints.

The problem with the constrained clustering is that most of the proposed methods are *ad hoc* in two ways. First, the objective of clustering is selected in an ad hoc way without a clear justification. Second, the optimization problem is usually NP-hard, and only heuristics are used to solve the problem.

2.2.2 Metric Learning for Clustering

Another approach—which is relevant to our representation learning approach in Chapter 3—keeps the clustering method fixed and instead searches for a metric that roughly fits the given constraints. In particular, the metric is learned based on some objective function over metrics [87, 5, 80], so that pairs of instances marked as *must-link* will be close in the new metric space (while *cannot-link* pairs are kept far apart).

Note, however, that the objective functions used for metric learning are rather *ad hoc*, and oblivious to the choice of clustering algorithm. In other words, it is not clear in what sense they are compatible with the adopted clustering algorithm (such as k-means). This means that performing clustering in the new space does not necessarily result in a clustering consistent with the given side-information.

A systematic way to define the objective of metric learning is to use the clustering loss directly. We will elaborate on this approach when we introduce the representation learning framework in Chapter 3.

Another way to address this deficiency is to combine the two optimization problems: the metric learning, and the constrained clustering. Bilenko et al. [31] proposed an objective function to optimize the metric and the clustering at the same time. They then used an iterative EM-type algorithm for optimization. Also, Basu et al. [23] proposed a similar framework with a different objective. The drawbacks of these integrated models are similar to those of constrained clustering: (i) the choice of objective function is not justified, and (ii) the proposed algorithm is not guaranteed to find a good solution to the optimization problem.

Assuming a probabilistic generative model for the data, Gopal and Yang [51] propose to learn a linear embedding of the data that is aligned with the labeled examples. We will elaborate more on this method in the next subsection.

2.2.3 Generative Models

Generative models are being used in different learning tasks, including semi-supervised clustering. In these models, it is assumed that the instances (together with their true assigned partitions) are generated from a structured distribution. The task is then to approximate this distribution based on the given labeled and unlabeled instances. In order to make this possible, one needs to make assumptions about the distribution of the data. The common approach is to consider a parametric class of distributions and try to estimate the parameters of the underlying distribution.

Basu et al. [24] considered a generative model based on Hidden Markov Random Fields (HMRFs). They showed that this model can be regarded as a probabilistic interpretation of [23], where the Euclidean distortion is generalized to Bregman's divergence. It was then showed [59] that this in turn is a special case of the weighted kernel k-means problem [40].

In a related work, Gopal and Yang [51] proposed an approach in which it is assumed that the data is generated by a mixture model (Gaussian or Von-Mises Fisher). The parameters of this model is then found such that the probability of generating the supervised labels is maximized. In particular, a shared covariance matrix is learned for all of the components (which is equivalent to learning a linear transformation for the data with unit variance model), enabling them to find clusters that were not present in the given supervised data.

These models are useful when we have solid information about the data generating distribution. However, in practice, the data is almost never generated exactly from the probabilistic model of choice. One way to address this situation is to provide an "agnostic" guarantee: the outcome of the algorithm should not be too bad if the assumption about the data generating distribution is 'marginally' violated. Unfortunately, we are not aware of such guarantees in the context of semi-supervised clustering.

Moreover, to make the problem computationally tractable, these methods usually resort to the maximum likelihood principle (rather than e.g., using the fully Bayesian approach). However, there is no guarantee that the maximum likelihood solution would be desirable (as it uses only a point-estimate of the hidden variables). Finally, even finding the solution to the maximum likelihood problem is sometimes computationally hard.

Despite the shortcomings of the existing models, there is a great potential for the applicability of this general methodology as it provides a natural way of encoding domain knowledge for semi-supervised clustering.

2.2.4 The Merge-Split Model

In Section 2.1, we briefly mentioned the framework proposed by Balcan and Blum [18]. In this setting (which is the first interactive clustering model that is formally analyzed), the learner outputs a clustering in each step, and the teacher corrects him by advising to either merge two clusters or split a cluster. In the beginning, the only thing that the learner knows is that the true clustering belongs to a given set of possible clusterings (i.e., a hypothesis class). In [18, 17], the computational and query complexity of this problem was investigated, showing some upper and lower bounds (e.g., for the case of finite hypothesis classes).

In order to get those bounds, it is expected from the teacher to respond to queries that have an excessive size (i.e., the teacher needs to look at the whole clustering of the data each time)—a task that is often exhausting (if not impossible) for the teachers. Therefore, this framework is especially applicable for the cases where the outcome of the clustering can be visualized and comprehended by a domain expert. Some efforts have been made to extend this framework to more practical scenarios, e.g., by considering the case of noisy teachers or a teachers with incomplete response [17].

Another issue that may arise is that the outcome of clustering can drastically change in each iteration, making it hard for the user to understand and guide the outcome of the algorithm. In order to handle this, [15] considered a setting where in each iteration only local changes are made to the clustering outcome.

The interactive nature of the merge-split model is particularly interesting. Also, the framework is theoretically solid and the algorithms are accompanied with theoretical guarantees of success. Note, however, that currently the positive results are proved only for some special hypothesis classes; also, the provided algorithms are usually not sufficiently efficient for practical applications. Improving these results is therefore an important direction for future research.

2.2.5 Property-based Clustering

A totally different approach to the problem of communicating user expertise for the purpose of choosing a clustering tool is discussed in [2]. They considered a set of *properties* (or *requirements*) for clustering algorithms, and investigated which of those properties hold for various algorithms. The user can then pick the right algorithm based on the requirements that she wants the algorithm to meet.

However, to turn such an approach into a practically useful tool, one will need to come up with properties that are relevant to the end user of clustering – a goal that is still far from being reached. Also, these properties are more useful for picking the general clustering paradigm (e.g., agglomerative or center-based), rather than picking the specific parameters (e.g, the target embedding).

2.3 Conclusions

We reviewed the existing methods for semi-supervised clustering. In each of these models, the domain knowledge is conveyed, modeled, and then used by the clustering algorithm in a certain way. Many of these methods, however, follow a rather ad-hoc approach that is not theoretically justified. In the following, we mention some of the drawbacks of the existing approaches.

- **The choice of the objective function is not justified.** For the metric learning methods, the objective function is usually picked in a way that makes the optimization problem easy to solve. However, it is not clear why optimizing such an objective would translate into a desirable clustering. For the constrained clustering methods, usually the objective function is not even explicit, and it is not clear how the outcome of the iterative method should be interpreted/evaluated. Finally, as described before, the use of maximum likelihood approach for the generative models is not well justified.

- **It is not clear how much 'advice' is needed.** It is important to know how much advice (e.g., constraints, queries, etc.) from the domain expert is required to make semi-supervised clustering possible. With the exception of the merge-split model [18], no upper or lower bounds for the advice complexity of the existing methods is obtained.

- **The objective functions are hard to optimize.** Adding constraints to the optimization problems usually makes them harder to solve. For example, the unconstrained version of the k-means clustering problem is already NP-hard, and therefore the constrained version is even harder to tackle. This is usually the case for generative models (specially in the fully Bayesian setup) as well. Furthermore, clustering in the merge-split model can be computationally hard too.

- **The supervision protocol is not user friendly.** In the merge-split model, it is required for the domain expert to analyze the clustering of the whole data set every time a query is asked. Aside from certain applications where the clustering of the data is easily visualizable/interpretable, this task is impossible for the user of the clustering method.

- **The assumptions about the data are not realistic.** In generative models for clustering, it is usually assumed that the data is generated by a specific parametric distribution. Unfortunately, there is no guarantee about the outcome of these methods when the true distribution fails to match the expectations. In particular, it is important to have a robust algorithm with an 'agnostic' guarantee.

It is of course hard to address all of these issues in a single unified framework. However, there is still much room for improvement, particularly in the development of the theoretical aspects of the semi-supervised clustering problem.

Chapter 3

Representation Learning for Clustering with Advice

The aim of Clustering with Advice (CLAD) is developing a systematic approach to convey and utilize domain knowledge for clustering applications. In particular, we are looking for a semi-supervised clustering framework, where the supervised feedback can be used to perform model selection for clustering. Therefore, defining a supervision protocol and a learning model is essential in enabling CLAD.

In this chapter, we approach the challenge by considering a scenario in which the domain expert (i.e., the intended user of the clustering) conveys her domain knowledge by providing a clustering of a small random subset of her data set. For example, consider a big customer service center that wishes to cluster incoming requests into groups to streamline their handling. Since the data base of requests is too large to be organized manually, the service center wishes to employ a clustering program. As the clustering designer, we would then ask the service center to pick a random sample of requests, manually cluster them, and show us the resulting grouping of that sample. The learning algorithm then uses that demonstration to pick a clustering method that, when applied to the full data set, will result in a clustering that follows the patterns demonstrated by that sample clustering. We address this paradigm from a statistical machine learning perspective.

Aiming to achieve generalization guaranties for such an approach, it is essential to introduce some *inductive bias*. We do that by restricting the clustering algorithm to a predetermined hypothesis class (or a set of concrete clustering algorithms). In a recent Dagstuhl workshop, Blum [32] proposed to do that by fixing a clustering algorithm, say k-means, and searching for a metric over the data under which k-means optimization yields a clustering that agrees with the training sample clustering. One should note that, given any domain set X, for any k-partitioning P of X, there exists some distance function d_P over X such that P is the optimal k-means clustering solution to the input (X, d_P)[1]. Consequently, to protect against potential overfitting, the class of potential distance functions should be constrained. In this chapter, we provide (apparently the first) concrete formal framework for such a paradigm, as well as a generalization analysis of this approach.

In this work we focus on center based clustering – an important class of clustering algorithms. In these algorithms, the goal is to find a set of "centers" (or prototypes), and the clusters are the Voronoi cells induced by this set of centers. The objective of such a clustering is to minimize the expected value of some monotonically increasing function of the distances of points to their cluster centers. The k-means clustering objective is arguably the most popular clustering paradigm in this class. Currently, center-based clustering tools lack a vehicle for incorporating domain expertise. Domain knowledge is usually taken into account only through an ad hoc choice of input data representation. Regretfully, it might not be realistic to require the domain expert to translate sufficiently elaborate task-relevant knowledge into hand-crafted features.

As a model for learning representations, we assume that the user-desirable clustering can be approximated by first mapping the sample to some Euclidean (or Hilbert) space and then performing k-means clustering in the mapped space (or equivalently, replacing the input data metric by some kernel and performing center-based clustering with respect to that kernel). Here, the clustering algorithm is supposed to learn a suitable mapping based on the given sample clustering. We call this approach *ReCLAD* which stands for *REpresentation learning for CLustering with ADvice*.

The main question addressed in this chapter is that of the sample complexity: what is the size of a sample, to be clustered by the domain expert, that suffices for finding a close-to-optimal mapping (i.e., a mapping that generalizes well on the test data)? Intuitively, this sample complexity depends on the richness of the class of potential mappings that the algorithm is choosing from. In standard supervised learning, there are well established notions of capacity of hypothesis classes (e.g., VC-dimension) that characterize the sample complexity of learning. This chapter aims to provide such relevant notions of capacity for clustering.

3.1 Contributions

Our first contribution is to provide a statistical framework to analyze the problem of representation learning for clustering. We assume that the expert has some implicit target clustering of the dataset

[1] This property is sometimes called k-Richness

in his mind. The learner however, is unaware of it, and instead has to select a mapping among a set of potential mappings, under which the result of k-means clustering will be similar to the target partition. An appropriate notion of loss function is introduced to quantify the success of the learner. Then, we define the analogous notion of PAC-learnability[2] for the problem of learning representation for clustering.

The second contribution of this chapter is the introduction of a combinatorial parameter, a specific notion of the capacity of the class of mappings, that determines the sample complexity of the clustering learning tasks. This combinatorial notion is a multivariate version of *pseudo-dimension* of a class of real-valued mappings. We show that there is *uniform convergence* of empirical losses to the true loss, over any class of embeddings, $\mathcal{F}$, at a rate that is determined by the proposed dimension of $\mathcal{F}$. This implies that any empirical risk minimization algorithm (ERM) will successfully learn such a class from sample sizes upper bounded by those rates. Finally, we analyze a particular natural class—the class of linear mappings from $\mathbb{R}^{d_2}$ to $\mathbb{R}^{d_1}$—and show that roughly speaking, sample size of $O(\frac{d_1 d_2}{\epsilon^2})$ is sufficient to guarantee an ϵ-optimal representation.

3.2 Preliminaries and Notations

Let X be a finite domain set. A *k-clustering* of X is a partition of X into k subsets. If C is a k-clustering, we denote the subsets of the partition by $C_1, ..., C_k$, therefore we have $C = \{C_1, .., C_k\}$. Let π^k denote the set of all permutations over $[k]$ where $[k]$ denotes $\{1, 2, ..., k\}$. We define the difference between two k-clusterings, C^1 and C^2, with respect to X as follows

$$\Delta_X(C^1, C^2) = \min_{\sigma \in \pi^k} \frac{1}{|X|} \sum_{i=1}^{k} |C_i^1 \Delta C_{\sigma(i)}^2| \tag{3.1}$$

where $|.|$ and Δ denote the cardinality and the symmetric difference of sets respectively. For a sample $S \subset X$, and C^1 (a partition of X), we define $C^1\Big|_S$ to be a partition of S induced by C^1, namely $C^1\Big|_S = \{C_1^1 \cap S, \ldots, C_k^1 \cap S\}$. Accordingly, the sample-based difference between two partitions is defined by

$$\Delta_S(C^1, C^2) = \Delta_S(C^1\Big|_S, C^2\Big|_S) \tag{3.2}$$

[2] PAC stands for the well known notion of “probably approximately correct”, popularized by [81].

Fix an unsupervised clustering algorithm, e.g., k-means clustering, that given a data set, outputs a k-partition of the data. We denote C_X as the outcome of clustering X (i.e., it is a k-clustering of X). Note that the unsupervised clustering algorithm is fixed and should be clear from the context.

Let f be a mapping from X to $\mathbb{R}^d$. We define C_X^f the result of clustering X after mapping it to a new space using f. In other words, $C_X^f = C_{f(X)}$.

The difference between two mappings f_1 and f_2 with respect to X is defined by the difference between the result of clustering using these mappings. Formally,

$$\Delta_X(f_1, f_2) = \Delta_X(C_X^{f_1}, C_X^{f_2}) \tag{3.3}$$

3.3 Formal Problem Statement (PAC-ReCLAD)

Let C^* be the target k-clustering of X. A *representation learning algorithm* $A(.,.)$ takes as input a sample set $S \subset X$ and its clustering, $C^*\Big|_S$, and outputs a mapping f from a set of mappings $\mathcal{F}$.

We call this learning problem ReCLAD which stands for *REpresentation learner for CLustering with ADvice*.

Definition 3.1. *Probably Approximately Correct Representation Learning for Clustering with Advice (PAC-ReCLAD)*

Let $\mathcal{F}$ be a set of mappings from X to $\mathbb{R}^d$. A representation learning algorithm A is a PAC-ReCLAD learner with sample complexity $m_{\mathcal{F}} : (0,1)^2 \mapsto \mathbb{N}$ with respect to $\mathcal{F}$, if for every $(\epsilon, \delta) \in (0,1)^2$, every domain set X and every clustering of X, C^*, the following holds:

if S is a randomly (uniformly) selected subset of X of size at least $m_{\mathcal{F}}(\epsilon, \delta)$, then with probability at least $1 - \delta$

$$\Delta_X(C^*, C_X^{f_A}) \leq \inf_{f \in \mathcal{F}} \Delta_X(C^*, C_X^f) + \epsilon \tag{3.4}$$

where $f_A = A(S, C^*\Big|_S)$, is the output of the algorithm.

Remark 3.1. *In this definition, f_A is the mapping that the algorithm outputs. Using this mapping, X is mapped to a new space. The result of clustering in this new space is $C_X^{f_A}$. Therefore, it is assumed that a fixed unsupervised clustering is used to cluster the data in the new space. In the next section, we will fix the k-means clustering for this purpose.*

Remark 3.2. *This can be regarded as a formal PAC framework to analyze the problem of clustering with advice. The learner is compared to the best mapping in the class $\mathcal{F}$. This means that this is an* agnostic *framework.*

Remark 3.3. *In this proposal, we investigate the* transductive *setup, where there is a given data set, known to the learner, that needs to be clustered. Clustering often occurs as a task over some data generating distribution (e.g., [85]). The current work can be readily extended to that setting. However, in that case, we assume that the clustering algorithm gets, on top of the clustered sample, a large unclustered sample drawn form that data generating distribution.*

A natural question is providing bounds on the sample complexity of PAC-ReCLAD with respect to $\mathcal{F}$. Intuitively, for richer classes of mappings, we need larger clustered samples. Therefore, we need to introduce an appropriate notion of "capacity" for $\mathcal{F}$ and bound the sample complexity based on it. This is addressed in the next sections.

In the next section, we specialize the general framework of ReCLAD for the case of k-means clustering.

3.4 The Case of K-means Clustering (ReKLAD)

In the previous section it was stated that the ReCLAD method relies on an unsupervised clustering method. In this section, we fix the k-means clustering algorithm in the ReCLAD framework. It means that we are looking for a representation of data under which the result of k-means clustering is consistent with the domain knowledge. We call this approach ReKLAD (which stands for REpresentation Learning for K-means clustering with ADvice).

k-means is a center-based clustering method. This means that the clustering outcome is the Voronoi cells induces by the set of k centers that the algorithm outputs. The k-means clustering objective is arguably the most popular center-based clustering paradigm. This makes the study of ReKLAD interesting and important. Also, k-means is especially interesting because it is flexible: for *any* target clustering in any domain, there exists a corresponding embedding to a new space such that the solution of k-means in the new space is the same as target clustering[3].

We formulate the ReCLAD problem for the case of k-means clustering. In the following, we introduce the formal definitions.

[3] This property is sometimes called k-Richness

3.4.1 Definitions and Notations

Let f be a mapping from X to $\mathbb{R}^d$, and $\mu = (\mu_1, \dots \mu_k)$ be a vector of k centers in $\mathbb{R}^d$. The clustering defined by (f, μ) is the partition over X induced by the μ-Voronoi partition in $\mathbb{R}^d$. Namely,

$$C_f(\mu) = (C_1, \dots C_k), \text{ where for all } i,$$
$$C_i = \{x \in X : \|f(x) - \mu_i\|_2 \le \|f(x) - \mu_j\|_2 \text{ for all } j \neq i\}$$

The k-means cost of clustering X with a set of centers $\mu = \{\mu_1, \dots, \mu_k\}$ and with respect to a mapping f is defined by

$$COST_X(f, \mu) = \frac{1}{|X|} \sum_{x \in X} \min_{\mu_i \in \mu} \|f(x) - \mu_i\|_2^2 \tag{3.5}$$

The k-means clustering algorithm finds the set of centers μ_X^f that minimize this cost[4]. In other words,

$$\mu_X^f = \arg\min_{\mu} COST_X(f, \mu) \tag{3.6}$$

Also, for a partition C and mapping f, we can define the cost of clustering as follows.

$$COST_X(f, C) = \frac{1}{|X|} \sum_{i \in [k]} \min_{\mu_j} \sum_{x \in C_i} \|f(x) - \mu_j\|_2^2 \tag{3.7}$$

The following proposition shows the "k-richness" property of k-means objective.

Proposition 3.1. *Let X be a domain set. For every k-clustering of X, C, and every $d \in \mathbb{N}^+$, there exist a mapping $g : X \mapsto \mathbb{R}^d$ such that $C_X^g = C$.*

Proof. The mapping g can be picked such that it collapses each cluster C_i into a single point in $\mathbb{R}^n$ (and so the image of X under mapping g will be just k single points in $\mathbb{R}^n$). The result of k-means clustering under such mapping will be C. □

For a mapping f as above, let C_X^f denote the k-means clustering of X induced by f, namely

$$C_X^f = C_f(\mu_X^f) \tag{3.8}$$

[4] We assume that the solution to k-means clustering is unique. We will elaborate about this issue in the next sections.

3.4.2 PAC-ReKLAD

Now that we have the needed notations, we can formally define the PAC-ReKLAD problem. However, the definition is exactly the same as that of PAC-ReCLAD (Definition 3.1). We only need to make the use of k-means clustering as the unsupervised tool explicit.

We avoid repeating the definition. We just note for PAC-ReKLAD is the same as PAC-ReCLAD, except that the meaning of C_X^f is more explicit: C_X^f is k-clustering induced by first mapping X to a new space using f, and then performing *k-means* clustering in the new space.

Proving a bound on the sample complexity of PAC-ReKLAD is the subject of the rest of this chapter.

3.5 Statistical Analysis of ReKLAD

The important question that was raised in the previous sections was that of the sample complexity (i.e., advice complexity): what is the size of a sample, to be clustered by the domain expert, that suffices for finding a close-to-optimal embedding (i.e., a mapping that generalizes well on the test data)?

Intuitively, this sample complexity depends on the richness of the class of potential embeddings that the algorithm is choosing from. In standard supervised learning, there are well established notions of capacity of hypothesis classes (e.g., VC-dimension) that characterize the sample complexity of learning. In this chapter we will introduce relevant notions of capacity for ReCLAD.

Particularly, we introduce a combinatorial parameter, a specific notion of the capacity of the class of mappings, that determines the advice complexity of ReKLAD. This combinatorial notion is a multivariate version of *pseudo-dimension* of a class of real-valued mappings. We show that there is *uniform convergence* of empirical losses to the true loss, over any class of mappings, $\mathcal{F}$, at a rate that is determined by the proposed dimension.

This implies that any empirical risk minimization algorithm (ERM) will successfully learn such a class from sample sizes upper bounded by those rates.

Finally, we analyze a particular natural class – the class of linear mappings from $\mathbb{R}^{d_2}$ to $\mathbb{R}^{d_1}$ – and show that roughly speaking, sample size of $O(\frac{d_1 d_2}{\epsilon^2})$ is sufficient to guarantee an ϵ-optimal answer.

3.5.1 Technical Background

Statistical convergence rates of sample clustering loss to the optimal clustering loss, with respect to some data generating probability distribution, play a central role in our analysis. From that perspective, most relevant to our work in this chapter are results that provide generalization bounds for k-means clustering. Ben-David [27] proposed the first dimension-independent generalization bound for k-means clustering loss based on compression techniques. This result was tightened in [30] through an analysis of Rademacher complexity. Also, [69] investigated a more general framework, in which generalization bounds for k-means as well as other algorithms can be obtained.

It should be noted that these results are about the standard clustering setup (without any supervised feedback), where the data representation is fixed and known to the clustering algorithm. However, analysis of the semi-supervised clustering problem – particularly PAC-ReKLAD – requires new tools. Also, note that the loss function in ReKLAD is not the usual k-means clustering loss.

3.5.2 ERM as a Representation Learner

In order to prove an upper bound for the sample complexity of ReKLAD, we need to consider an algorithm, and prove a sample complexity bound for it. Here, we show that any ERM-type algorithm[5] can be used for the ReKLAD framework. Therefore, we will be able to prove an upper bound for the sample complexity of PAC-ReKLAD.

Let $\mathcal{F}$ be a class of mappings and X be the domain set. A TERM[6] learner for $\mathcal{F}$ takes as input a sample $S \subset X$ and its clustering Y and outputs:

$$A^{TERM}(S, Y) = \operatorname*{arg\,min}_{f \in \mathcal{F}} \Delta_S(C_X^f\Big|_S, Y) \tag{3.9}$$

Note that we call it transductive, because it is implicitly assumed that it has access to the unlabeled dataset (i.e., X). A TERM algorithm goes over all mappings in $\mathcal{F}$ and selects the mapping which is the most consistent mapping with the given clustering: the mapping under which if we perform k-means clustering of X, the sample-based Δ-difference between the result and Y is minimized.

[5] ERM stands for Empirical Risk Minimization

[6] TERM stands for Transductive Empirical Risk Minimizer

Intuitively, this algorithm will work well when the empirical Δ-difference and the true Δ-difference of the mappings in the class are close to each other. In this case, by minimizing the empirical difference, the algorithm will automatically minimize the true difference as well. In order to formalize this idea, we define the notion of "representativeness" of a sample.

Definition 3.2. *(ϵ-Representative Sample) Let $\mathcal{F}$ be a class of mappings from X to $\mathbb{R}^d$. A sample S is ϵ-representative with respect to $\mathcal{F}$, X and the clustering C^*, if for every $f \in \mathcal{F}$ the following holds*

$$|\Delta_X(C^*, C_X^f) - \Delta_S(C^*, C_X^f))| \leq \epsilon \tag{3.10}$$

The following theorem shows that for the TERM algorithm to work, it is sufficient to supply it with a representative sample.

Theorem 3.1. *(Sufficiency of Uniform Convergence) Let $\mathcal{F}$ be a set of mappings from X to $\mathbb{R}^d$. If S is an $\frac{\epsilon}{2}$-representative sample with respect to X, $\mathcal{F}$ and C^* then*

$$\Delta_X(C^*, C_X^{\hat{f}}) \leq \Delta_X(C^*, C_X^{f^*}) + \epsilon \tag{3.11}$$

where $f^* = \arg\min_{f \in \mathcal{F}} \Delta_X(C^*, C_X^f)$ and $\hat{f} = A^{TERM}(S, C^*\Big|_S)$.

Proof. Using $\frac{\epsilon}{2}$-representativeness of S and the fact that $\hat{f}$ is the empirical minimizer of the loss function, we have

$$\Delta_X(C^*, C_X^{\hat{f}}) \leq \Delta_S(C^*, C_X^{\hat{f}}) + \frac{\epsilon}{2} \tag{3.12}$$

$$\leq \Delta_S(C^*, C_X^{f^*}) + \frac{\epsilon}{2} \tag{3.13}$$

$$\leq \Delta_X(C^*, C_X^{f^*}) + \frac{\epsilon}{2} + \frac{\epsilon}{2} \tag{3.14}$$

$$\leq \Delta_X(C^*, C_X^{f^*}) + \epsilon \tag{3.15}$$

□

Therefore, we just need to provide an upper bound for the sample complexity of uniform convergence: "how many instances do we need to make sure that with high probability our sample is ϵ-representative?"

3.5.3 Classes of Mappings with a Uniqueness Property

In general, the solution to k-means clustering may not be unique. Therefore, the learner may end up with finding a mapping that corresponds to multiple different clusterings. This is not desirable, because in this case, the output of the learner will not be interpretable. Therefore, it is reasonable to choose the class of potential mappings in a way that it includes only the mappings under which the solution is unique.

In order to make this idea concrete, we need to define an appropriate notion of uniqueness. We use a notion similar to the one introduced by [19] with a slight modification[7].

Definition 3.3. *((η, ϵ)-Uniqueness) We say that k-means clustering for domain X under mapping $f : \mathcal{X} \mapsto \mathbb{R}^d$ has a (η, ϵ)-unique solution, if every η-optimal solution of the k-means cost is ϵ-close to the optimal solution. Formally, the solution is (η, ϵ)-unique if for every partition P that satisfies*

$$COST_X(f, P) < COST_X(f, C_X^f) + \eta \tag{3.16}$$

would also satisfy

$$\Delta_X(C_X^f, P) < \epsilon \tag{3.17}$$

In the degenerate case where the optimal solution to k-means is not unique itself (and so C_X^f is not well-defined), we say that the solution is not (η, ϵ)-unique.

It can be noted that the definition of (η, ϵ)-uniqueness not only requires the optimal solution to k-means clustering to be unique, but also all the "near-optimal" minimizers of the k-means clustering cost should be "similar". This is a natural strengthening of the uniqueness condition, to guard against cases where there are η_0-optimizers of the cost function (for arbitrarily small η_0) with totally different solutions.

Now that we have a definition for uniqueness, we can define the set of mappings for X under which the solution is unique. We say that a class of mappings F has (η, ϵ)-*uniqueness property* with respect to X, if every mapping in F has (η, ϵ)-uniqueness property over X.

Note that given an arbitrary class of mappings F, we can find a subset of it that satisfies (η, ϵ)-uniqueness property over X. Also, as argued above, this subset is the useful subset to work with. Therefore, in the rest of this chapter, we investigate learning for classes with (η, ϵ)-uniqueness property. In the next section, we prove uniform convergence results for such classes.

[7] Our notion is additive in both parameters rather than multiplicative

3.6 Uniform Convergence Results

In Section 3.5.2, we defined the notion of ϵ-representative samples. Also, we proved that if a TERM algorithm is fed with such a representative sample, it will work satisfactorily. The most technical part of the proof is then about the question “how large should be the sample in order to make sure that with high probability it is actually a representative sample?”

In order to formalize this notion, let $\mathcal{F}$ be a set of mappings from a domain X to $(0,1)^{n}$[8]. Define the sample complexity of uniform convergence, $m_{\mathcal{F}}^{UC}(\epsilon,\delta)$, as the minimum number m such that for every fixed partition C^*, if S is a randomly (uniformly) selected subset of X with size m, then with probability at least $1-\delta$, for all $f \in \mathcal{F}$ we have

$$|\Delta_X(C^*, C_X^f) - \Delta_S(C^*, C_X^f)| \leq \epsilon \tag{3.18}$$

The technical part of this chapter is devoted to provide an upper bound for this sample complexity.

3.6.1 Preliminaries

Definition 3.4. *(ϵ-cover and covering number) Let $\mathcal{F}$ be a set of mappings from X to $(0,1)^n$. A subset $\hat{F} \subset \mathcal{F}$ is called an ϵ-cover for $\mathcal{F}$ with respect to the metric $d(.,.)$ if for every $f \in \mathcal{F}$ there exists $\hat{f} \in \hat{F}$ such that $d(f,\hat{f}) \leq \epsilon$. The covering number, $\mathcal{N}(\mathcal{F}, d, \epsilon)$ is the size of the smallest ϵ-cover of $\mathcal{F}$ with respect to d.*

In the above definition, we did not specify the metric d. In our analysis, we are interested in the L_1 distance with respect to X, namely:

$$d_{L_1}^X(f_1, f_2) = \frac{1}{|X|}\sum_{x \in X} \|f_1(x) - f_2(x)\|_2 \tag{3.19}$$

Note that the mappings we consider are not real-valued functions, but their output is an n-dimensional vector. This is in contrast to the usual analysis used for learning real-valued functions. If f_1 and f_2 are real-valued, then L_1 distance is defined by

[8] In the analysis, for simplicity, we will assume that the set of mappings is a function to the bounded space $(0,1)^n$ wherever needed

$$d_{L_1}^X(f_1, f_2) = \frac{1}{|X|} \sum_{x \in X} |f_1(x) - f_2(x)| \tag{3.20}$$

We will prove sample complexity bounds for our problem based on the L_1-covering number of the set of mappings. However, it will be beneficial to have a bound based on some notion of capacity, similar to VC-dimension, as well. This will help in better understanding and easier analysis of sample complexity of different classes. While VC-dimension is defined for binary valued functions, we need a similar notion for functions with outputs in $\mathbb{R}^n$. For real-valued functions, we have such notion, called pseudo-dimension [78].

Definition 3.5. *(Pseudo-Dimension) Let $\mathcal{F}$ be a set of functions from X to $\mathbb{R}$. Let $S = \{x_1, x_2, \ldots, x_m\}$ be a subset of X. Then S is pseudo-shattered by $\mathcal{F}$ if there are real numbers $r_1, r_2, \ldots, r_m$ such that for every $b \in \{0,1\}^m$, there is a function $f_b \in \mathcal{F}$ with $sgn(f_b(x_i) - r_i) = b_i$ for $i \in [m]$. Pseudo dimension of $\mathcal{F}$, called $Pdim(\mathcal{F})$, is the size of the largest shattered set.*

It can be shown (e.g., Theorem 18.4. in [8]) that for a real-valued class F, if $Pdim(F) \leq q$ then $\log \mathcal{N}(F, d_{L_1}^X, \epsilon) = \mathcal{O}(q)$ where $\mathcal{O}()$ hides logarithmic factors of $\frac{1}{\epsilon}$. In the next sections, we will generalize this notion to $\mathbb{R}^n$-valued functions.

3.6.2 Reduction to Binary Hypothesis Classes

Let $f_1, f_2 \in \mathcal{F}$ be two mappings and σ be a permutation over $[k]$. Define the binary-valued function $h_\sigma^{f_1,f_2}(.)$ as follows

$$h_\sigma^{f_1,f_2}(x) = \begin{cases} 1 & x \in \cup_{i=1}^k (C_i^{f_1} \Delta C_{\sigma(i)}^{f_2}) \\ 0 & \text{otherwise} \end{cases} \tag{3.21}$$

Let $H_\sigma^{\mathcal{F}}$ be the set of all such functions with respect to $\mathcal{F}$ and σ:

$$H_\sigma^{\mathcal{F}} = \{h_\sigma^{f_1,f_2}(.) : f_1, f_2 \in \mathcal{F}\} \tag{3.22}$$

Finally, let $H^{\mathcal{F}}$ be the union of all $H_\sigma^{\mathcal{F}}$ over all choices of σ. Formally, if π is the set of all permutations over $[k]$, then

$$H^{\mathcal{F}} = \cup_{\sigma \in \pi} H_\sigma^{\mathcal{F}} \tag{3.23}$$

For a set S, and a binary function $h(.)$, let $h(S) = \frac{1}{|S|}\sum_{x\in S} h(x)$. We now show that a uniform convergence result with respect to $H^{\mathcal{F}}$ is sufficient to have uniform convergence for the Δ-difference function. Therefore, we will be able to investigate conditions for uniform convergence of $H^{\mathcal{F}}$ rather than the Δ-difference function.

Theorem 3.2. *Let X be a domain set, $\mathcal{F}$ be a set of mappings, and $H^{\mathcal{F}}$ be defined as above. If $S \subset X$ is such that*

$$\forall h \in H^{\mathcal{F}}, |h(S) - h(X)| \leq \epsilon \tag{3.24}$$

then S will be 2ϵ-representative with respect to $\mathcal{F}$, i.e., for all $f_1, f_2 \in \mathcal{F}$ we will have

$$|\Delta_X(C_X^{f_1}, C_X^{f_2}) - \Delta_S(C_X^{f_1}, C_X^{f_2})| \leq 2\epsilon \tag{3.25}$$

Proof.

$$|\Delta_S(C_X^{f_1}, C_X^{f_2}) - \Delta_X(C_X^{f_1}, C_X^{f_2})| \tag{3.26}$$

$$= \left| \left(\min_\sigma \frac{1}{|S|} \sum_{x\in S} h_\sigma^{f_1,f_2} \right) - \left(\min_\sigma \frac{1}{|X|} \sum_{x\in X} h_\sigma^{f_1,f_2} \right) \right| \tag{3.27}$$

$$\leq 2 \left| \max_\sigma \left(\frac{1}{|S|} \sum_{x\in S} h_\sigma^{f_1,f_2} - \frac{1}{|X|} \sum_{x\in X} h_\sigma^{f_1,f_2} \right) \right| \tag{3.28}$$

$$\leq 2 \left| \max_\sigma \left(h_\sigma^{f_1,f_2}(S) - h_\sigma^{f_1,f_2}(X) \right) \right| \leq 2\epsilon \tag{3.29}$$

□

The fact that $H^{\mathcal{F}}$ is a class of binary-valued functions enables us to provide sample complexity bounds based on VC-dimension of this class. However, providing bounds based on VC-Dim$(H^{\mathcal{F}})$ is not sufficient, in the sense that it is not convenient to work with the class $H^{\mathcal{F}}$. Instead, it will be nice if we can prove bounds directly based on the capacity of the class of mappings, $\mathcal{F}$. In the next section, we address this issue.

3.6.3 L_1-Covering Number and Uniform Convergence

The classes introduced in the previous section, $H^{\mathcal{F}}$ and $H^{\mathcal{F}}_{\sigma}$, are binary hypothesis classes. Also, we have shown that proving a uniform convergence result for $H^{\mathcal{F}}$ is sufficient for our purpose. In this section, we show that a bound on the L_1 covering number of $\mathcal{F}$ is sufficient to prove uniform convergence for $H^{\mathcal{F}}$.

In Section 3.5.3, we argued that we only care about the classes that have (η, ϵ)-uniqueness property. In the rest of this section, assume that $\mathcal{F}$ is a class of mappings from X to $(0,1)^n$ that satisfies (η, ϵ)-uniqueness property.

Lemma 3.1. *Let $f_1, f_2 \in \mathcal{F}$. If $d_{L_1}(f_1, f_2) < \frac{\eta}{12}$ then $\Delta_X(f_1, f_2) < 2\epsilon$*

We leave the proof of this lemma for the appendix in the end of the chapter, and present the next lemma.

Lemma 3.2. *Let $H^{\mathcal{F}}$ be defined as in the previous section. Then,*

$$\mathcal{N}(H^{\mathcal{F}}, d^X_{L_1}, 2\epsilon) \leq k!\mathcal{N}(\mathcal{F}, d^X_{L_1}, \frac{\eta}{12}) \tag{3.30}$$

Proof. Let $\hat{\mathcal{F}}$ be the $\frac{\eta}{12}$-cover corresponding to the covering number $\mathcal{N}(\mathcal{F}, d^X_{L_1}, \frac{\eta}{12})$. Based on the previous lemma, $H^{\hat{\mathcal{F}}}_{\sigma}$ is a 2ϵ-cover for $H^{\mathcal{F}}_{\sigma}$. But we have only $k!$ permutations of $[k]$, therefore, the covering number for $H^{\hat{\mathcal{F}}}$ is at most $k!$ times larger than $H^{\hat{\mathcal{F}}}_{\sigma}$. This proves the result. □

Basically, this means that if we have a small L_1 covering number for the mappings, we will have the uniform convergence result we were looking for. The following theorem proves this result.

Theorem 3.3. *Let $\mathcal{F}$ be a set of mappings with (η, ϵ)-uniqueness property. Then for some constant $\alpha \geq 0$ we have*

$$m^{UC}_{\mathcal{F}}(\epsilon, \delta) \leq O(\frac{\log k! + \log \mathcal{N}(\mathcal{F}, d^X_{L_1}, \frac{\eta}{\alpha}) + \log(\frac{1}{\delta})}{\epsilon^2}) \tag{3.31}$$

Proof. Following the previous lemma, if we have a small L_1-covering number for $\mathcal{F}$, we will also have a small covering number for $H^{\mathcal{F}}$ as well. But based on standard uniform convergence theory, if a hypothesis class has small covering number, then it has uniform convergence property. More precisely, (e.g., Theorem 17.1 in [8]) we have:

$$m^{UC}_{H^{\mathcal{F}}}(\epsilon_0, \delta) \leq O(\frac{\log \mathcal{N}(H^{\mathcal{F}}, d^X_{L_1}, \frac{\epsilon_0}{16}) + \log(\frac{1}{\delta})}{\epsilon_0^2}) \tag{3.32}$$

Applying Lemma 3.2 to the above proves the result. □

3.6.4 Bounding L_1-Covering Number

In the previous section, we proved if the L_1-covering number of the class of mappings is bounded, then we will have uniform convergence. However, it is desirable to have a bound with respect to a combinatorial dimension of the class (rather than the covering number). Therefore, we will generalize the notion of pseudo-dimension for the class of mappings that take value in $\mathbb{R}^n$.

Let $\mathcal{F}$ be a set of mappings form X to $\mathbb{R}^n$. For every mapping $f \in \mathcal{F}$, define real-valued functions $f_1, \ldots, f_n$ such that $f(x) = (f_1(x), \ldots, f_n(x))$. Now let $F_i = \{f_i : f \in F\}$. This means that $F_1, F_2, \ldots, F_n$ are classes of real-valued functions. Now we define pseudo-dimension of $\mathcal{F}$ as follow.

$$Pdim(\mathcal{F}) = n \max_{i \in [n]} Pdim(F_i) \tag{3.33}$$

Proposition 3.2. *Let $\mathcal{F}$ be a set of mappings form X to $\mathbb{R}^n$. If $Pdim(F) \leq q$ then*

$$\log \mathcal{N}(F, d^X_{L_1}, \epsilon) = \mathcal{O}(q)$$

where $\mathcal{O}()$ hides logarithmic factors.

Proof. The result follows from the corresponding result for bounding covering number of real-valued functions based on pseudo-dimension mentioned in the preliminaries section. The reason is that we can create a cover by composition of the $\frac{\epsilon}{n}$-covers of all F_i. However, this will at most introduce a factor of n in the logarithm of the covering number. □

Therefore, we can rewrite the result of the previous section in terms of pseudo-dimension.

Theorem 3.4. *Let $\mathcal{F}$ be a class of mappings with (η, ϵ)-uniqueness property. Then*

$$m^{UC}_{\mathcal{F}}(\epsilon, \delta) \leq \mathcal{O}\left(\frac{k + Pdim(\mathcal{F}) + \log(\frac{1}{\delta})}{\epsilon^2}\right) \tag{3.34}$$

where $\mathcal{O}()$ hides logarithmic factors of k and $\frac{1}{\eta}$.

3.7 Sample Complexity of PAC-ReKLAD

In this section, we provide the main result of this chapter. In Section 3.5.2 we had showed that uniform convergence is sufficient for a TERM algorithm to work. Also, in the previous section, we proved a bound for the sample complexity of uniform convergence. The following theorem, which is the main technical result of this chapter, combines these two and provides a sample complexity upper bound for PAC-ReKLAD framework.

Theorem 3.5. (Sample Complexity of ReKLAD)

Let $\mathcal{F}$ be a class of (η, ϵ)-unique mappings. Then the sample complexity of representation learning for k-means clustering (ReKLAD) with respect to $\mathcal{F}$ is upper bounded by

$$m_{\mathcal{F}}(\epsilon, \delta) \leq \mathcal{O}(\frac{k + Pdim(\mathcal{F}) + \log(\frac{1}{\delta})}{\epsilon^2}) \tag{3.35}$$

where $\mathcal{O}$ hides logarithmic factors of k and $\frac{1}{\eta}$.

The proof is done by combining Theorems 3.1 and 3.4.

The following result shows an upper bound for the sample complexity of learning linear mappings (or equivalently, Mahalanobis metrics).

Corollary 3.1. *Let $\mathcal{F}$ be a set of (η, ϵ)-unique* linear *mappings from $\mathbb{R}^{d_1}$ to $\mathbb{R}^{d_2}$. Then we have*

$$m_{\mathcal{F}}(\epsilon, \delta) \leq \mathcal{O}(\frac{k + d_1 d_2 + \log(\frac{1}{\delta})}{\epsilon^2}) \tag{3.36}$$

Proof. It is a standard result that the pseudo-dimension of a vector space of real-valued functions is just the dimensionality of the space (in our case d_1) (e.g., Theorem 11.4 in [8]). Also, based on our definition of $Pdim$ for $\mathbb{R}^{d_2}$-valued functions, it should scale by a factor of d_2. □

3.8 Conclusions

In this chapter we introduced the problem of representation learning for clustering with advice (ReCLAD) and provided a formal statistical framework for analyzing this framework. In ReCLAD, the learner—unaware of the target clustering of the domain—is given a clustering of a small sample set. The learner's task is then finding a mapping (among a class of mappings) under which the result of clustering of the domain is as close as possible to the true clustering. For the special case of k-means clustering, this framework was called ReKLAD.

In section 3.5, we provided the results on the advice complexity of PAC-ReKLAD. More specifically, a notion of *vector-valued pseudo-dimension* for the class of mappings was defined, and the sample complexity was upper bounded based on it. This means that for the classes with higher such dimension, more clustered samples are required. Furthermore, it was proved that any ERM-type algorithm that has access to such a sample will work satisfactorily

In order to prove this result, a notion of uniform convergence was defined, and it was shown that the rate of convergence depends on the pseudo-dimension of the class of mappings. This was in turn proved using a bound on the covering number of the set of mappings.

3.8.1 Future Research Directions

The choice of k-means clustering was rather arbitrary, except that it is *rich*. Therefore, it will be useful to extend the results of PAC-ReKLAD to other clustering algorithms (i.e., considering the general PAC-ReCLAD framework).

It can be noted that we did not analyze the computational complexity of the proposed algorithms for PAC-ReKLAD. In fact, the problem is NP-hard, as the standard k-means clustering is hard even without learning the representation. However, it is important to provide computationally efficient algorithms. This can be done either by picking other clustering algorithms or by exploiting the "niceness" of data-generating distribution (e.g., a similar notion of uniqueness proposed by [19] makes the complexity of k-means clustering problem polynomial). In the next chapter, we propose another model for semi-supervised clustering where computationally efficient learning is actually possible.

In ReCLAD framework, we assumed that the clustered sample is picked randomly. However, we may also consider an *active/adaptive* setting, where the learner chooses this sample set gradually. Furthermore, we assumed that the number of clusters is given and fixed for both the main task (i.e., clustering of the whole domain set) and the clustering of the given sample. However, it is conceivable that the domain expert would partition the small sample into a fewer number of clusters. Therefore, it is important to "learn" how to pick the right number of clusters as well.

There are other supervision protocols that were discussed in Chapter 2. In particular, in many cases the supervised feedback is in the form of pairwise constraints. This is in contrast to CLAD framework where the domain expert gives the clustering of a random sample. Therefore, it is important to study the connection between these two scenarios, and possibly extend our results to the other case. Another supervision protocol which has not been studied yet is the *comparison-based* clustering where the domain expert is asked to compare two given clusterings and should select one that is better. This can be more intuitive for the expert in many cases.

Finally, in this framework, we used supervision as a tool to capture domain knowledge. However, in addition to the information-theoretic benefits of supervised feedback, there can be computational gains as well. For example, k-means clustering is NP-hard. However, if we have access to an oracle (i.e., domain expert), we may be able to find the solution using a few queries. We will study this new line of research in the next chapter.

3.9 Appendix: Proof of Lemma 3.1

Let $\mathcal{F} : X \mapsto (0,1)^n$ be a set of mappings that have (η, ϵ)-uniqueness property. Let $f_1, f_2 \in \mathcal{F}$ and $d_{L_1}(f_1, f_2) < \frac{\eta}{12}$. We need to prove that $\Delta_X(f_1, f_2) < 2\epsilon$. In order to prove this, note that due to triangular inequality, we have

$$\begin{aligned}\Delta_X(f_1, f_2) &= \Delta_X(C^{f_1}(\mu^{f_1}), C^{f_2}(\mu^{f_2})) \\ &\leq \Delta_X(C^{f_1}(\mu^{f_1}), C^{f_1}(\mu^{f_2})) + \Delta_X(C^{f_1}(\mu^{f_2}), C^{f_2}(\mu^{f_2}))\end{aligned} \tag{3.37}$$

Therefore, it will be sufficient to show that each of the Δ-terms above is smaller than ϵ. We start by proving a useful lemma.

Lemma 3.3. *Let $f_1, f_2 \in \mathcal{F}$ and $d_{L_1}(f_1, f_2) < \frac{\eta}{6}$. Let μ be an arbitrary set of k centers in $(0,1)^n$. Then*

$$|COST_X(f_1, \mu) - COST_X(f_2, \mu)| < \frac{\eta}{2}$$

Proof.

$$|COST_X(f_1, \mu) - COST_X(f_2, \mu)|$$

$$= \left| \left(\frac{1}{|X|} \sum_{x \in X} \min_{\mu_j \in \mu} \|f_1(x) - \mu_j\|^2 \right) - \left(\frac{1}{|X|} \sum_{x \in X} \min_{\mu_j \in \mu} \|f_2(x) - \mu_j\|^2 \right) \right| \tag{3.38}$$

$$\leq \frac{1}{|X|} \sum_{x \in X} \max_{\mu_j \in \mu} \left| \|f_1(x) - \mu_j\|^2 - \|f_2(x) - \mu_j\|^2 \right| \tag{3.39}$$

$$= \frac{1}{|X|} \sum_{x \in X} \max_{\mu_j \in \mu} \left| \|f_1(x)\|^2 - \|f_2(x)\|^2 - 2 < \mu_j, f_1 - f_2 > \right| \tag{3.40}$$

$$= \frac{1}{|X|} \sum_{x \in X} \max_{\mu_j \in \mu} \left| < f_1 - f_2, f_1 + f_2 - 2\mu_j > \right| \tag{3.41}$$

$$\leq \frac{3}{|X|}\sum_{x\in X}\|f_1 - f_2\| \leq \frac{3\eta}{6} \leq \frac{\eta}{2} \tag{3.42}$$

□

Now we are ready to prove that the first Δ-term is smaller than ϵ, i.e., $\Delta_X(C^{f_1}(\mu^{f_1}), C^{f_1}(\mu^{f_2})) < \epsilon$. But to do so, we only need to show that $COST_X(f_1, \mu^{f_2}) - COST_X(f_1, \mu^{f_1}) < \eta$; because in that case, due to (η, ϵ)-uniqueness property of f_1, the result will follow. Now, using Lemma 3.3, we have

$$COST_X(f_1, \mu^{f_2}) - COST_X(f_1, \mu^{f_1}) \tag{3.43}$$

$$\leq \Big(COST_X(f_2, \mu^{f_2}) + \frac{\eta}{2}\Big) - COST_X(f_1, \mu^{f_1}) \tag{3.44}$$

$$= \min_{\mu}(COST_X(f_2, \mu)) - \min_{\mu}(COST_X(f_1, \mu)) + \frac{\eta}{2} \tag{3.45}$$

$$\leq \max_{\mu}\,(COST_X(f_2, \mu) - COST_X(f_1, \mu)) + \frac{\eta}{2} \tag{3.46}$$

$$\leq \frac{\eta}{2} + \frac{\eta}{2} \leq \eta \tag{3.47}$$

where in the first and the last line we used Lemma 3.

Finally, we need to prove the second Δ-inequality, i.e., $\Delta_X(C^{f_1}(\mu^{f_2}), C^{f_2}(\mu^{f_2})) \leq \epsilon$. Assume contrary. But based on (η, ϵ)-uniqueness property of f_2, we conclude that $COST_X(f_2, C^{f_1}(\mu^{f_2})) - COST_X(f_2, C^{f_2}(\mu^{f_2})) \geq \eta$. In the following, we prove that this cannot be true, and hence a contradiction.

Let $m_x = \arg\min_{\mu_0 \in \mu^{f_2}} \|f_1(x) - \mu_0\|^2$. Then, based on the boundedness of $f_1(x)$,$f_2(x)$ and we have:

$$COST_X(f_2, C^{f_1}(\mu^{f_2})) - COST_X(f_2, C^{f_2}(\mu^{f_2})) \tag{3.48}$$

$$= \left(\frac{1}{|X|}\sum_{x\in X}\|f_2(x) - m_x\|^2\right) - COST_X(f_2, \mu_2) \tag{3.49}$$

$$= \left(\frac{1}{|X|}\sum_{x\in X}\|f_2(x) - f_1(x) + f_1(x) - m_x\|^2\right) - COST_X(f_2, \mu_2) \tag{3.50}$$

$$= \frac{1}{|X|}\sum_{x\in X}\|f_2(x)-f_1(x)\|^2 + \frac{1}{|X|}\sum_{x\in X}\|f_1(x)-m_x\|^2 \\ + \frac{1}{|X|}\sum_{x\in X} 2 < f_2(x)-f_1(x), f_1(x)-m_x > -COST_X(f_2,\mu_2) \tag{3.51}$$

$$\leq \frac{2}{|X|}\sum_{x\in X}\|f_2(x)-f_1(x)\| + COST_X(f_1,\mu_1) \\ + \frac{4}{|X|}\sum_{x\in X}\|f_2(x)-f_1(x)\| - COST_X(f_2,\mu_2) \tag{3.52}$$

$$\leq \frac{6}{|X|}\sum_{x\in X}\|f_2(x)-f_1(x)\| + (COST_X(f_1,\mu_1) - COST_X(f_2,\mu_2)) \tag{3.53}$$

$$\leq \frac{6\eta}{12} + \frac{\eta}{2} \leq \eta \tag{3.54}$$

Chapter 4

Efficient Clustering with Advice

Clustering is a challenging task particularly due to two impediments. The first problem is that clustering, in the absence of domain knowledge, is usually an *under-specified* task: the solution of choice may vary significantly between different intended applications. The second one is that performing clustering under many natural models is computationally hard.

Consider the task of dividing the users of an online shopping service into different groups. The result of this clustering can then be used for example in suggesting similar products to the users in the same group, or for organizing data so that it would be easier to read/analyze the monthly purchase reports. Those different applications may result in conflicting solution requirements. In such cases, one needs to exploit domain knowledge to better define the clustering problem. For example, the framework that we proposed in the previous chapter addressed the same problem.

At the same time, mitigating the computational problem of clustering is critical. Solving most of the common optimization formulations of clustering is NP-hard (in particular, solving the popular k-means and k-median clustering problems). One approach to address this issues is to exploit the fact that natural data sets usually exhibit some nice properties and likely to avoid the worst-case scenarios. In such cases, optimal solution to clustering may be found efficiently. The quest for notions of niceness that are likely to occur in real data and allow clustering efficiency is still ongoing (see [28] for a critical survey of work in that direction).

In this chapter, we take a new approach to alleviate the computational problem of clustering. In particular, we ask the following question: can weak supervision (in the form of answers to natural queries) help relaxing the computational burden of clustering? This will add up to the other benefit of supervision: making the clustering problem better defined by enabling the accession of domain knowledge through the supervised feedback.

The general setting considered in this chapter is the following. Let X be a set of elements that should be clustered and d a dissimilarity function over it. The oracle (e.g., a domain expert) has some information about a target clustering C^*_X in mind. The clustering algorithm has access to X, d, and can also make queries about C^*_X. The queries are in the form of *same-cluster* queries. Namely, the algorithm can ask whether two elements belong to the same cluster or not. The goal of the algorithm is to find a clustering that meets some predefined clusterability conditions and is consistent with the answers given to its queries.

We will also consider the case that the oracle conforms with some optimal k-means solution. We then show that access to a 'reasonable' number of same-cluster queries can enable us to provide an efficient algorithm for otherwise NP-hard problems.

4.1 Contributions

The two main contributions of this chapter are the introduction of the semi-supervised active clustering (SSAC) framework and, the rather unusual demonstration that access to simple query answers can turn an otherwise NP hard clustering problem into a feasible one.

Before we explain those results, let us also mention a notion of clusterability (or 'input niceness') that we introduce. We define a novel notion of niceness of data, called γ-margin property that is related to the previously introduced notion of center proximity [16]. The larger the value of γ, the stronger the assumption becomes, which means that clustering becomes easier. With respect to that γ parameter, we get a sharp 'phase transition' between k-means being NP hard and being optimally solvable in polynomial time[1].

We focus on the effect of using queries on the computational complexity of clustering. We provide a probabilistic polynomial time (BPP) algorithm for clustering with queries, that succeeds under the assumption that the input satisfies the γ-margin condition for $\gamma > 1$. This algorithm makes $O\big(k^2 \log k + k \log n\big)$ same-cluster queries to the oracle and runs in $O\big(kn \log n\big)$ time, where k is the number of clusters and n is the size of the instance set.

On the other hand, we show that without access to query answers, k-means clustering is NP-hard even when the solution satisfies γ-margin property for $\gamma = \sqrt{3.4} \approx 1.84$ and $k = \Theta(n^\epsilon)$ (for any $\epsilon \in (0,1)$). We further show that access to $\Omega(\log k + \log n)$ queries is needed to overcome the NP hardness in that case. These results, put together, show an interesting phenomenon. Assume that the oracle conforms to an optimal solution of k-means clustering and that it satisfies

[1] The exact value of such a threshold γ depends on some finer details of the clustering task; whether d is required to be Euclidean and whether the cluster centers must be members of X.

the γ-margin property for some $1 < \gamma \leq \sqrt{3.4}$. In this case, our lower bound means that without making queries k-means clustering is NP-hard, while the positive result shows that with a reasonable number of queries the problem becomes efficiently solvable.

This indicates an interesting trade-off between query complexity and computational complexity in the clustering domain.

4.2 Related Work

In Chapter 2 we reviewed the relevant literature to semi-supervised clustering. To recap, the most common method to convey supervision for clustering is through a set of pairwise *must-link/cannot-link* constraints on the instances [22, 24, 59]. Note that in contrast to the interactive supervision protocol that we will propose, the supervision is non-interactive in these scenarios. Another example of the non-interactive use of supervision was the ReCLAD framework introduced in the previous chapter.

On the theory side, Balcan et al. [18] proposed a framework for interactive clustering with the help of a user. In particular, the user was provided with the current clustering, and told the algorithm to either split a cluster or merge two clusters. See Section 2.2.4 for details.

Our proposed setup combines the user-friendliness of *must-link/cannot-link* queries (as opposed to asking the domain expert to answer queries about whole data set clustering, or to cluster sets of data) with the advantages of interactiveness.

Furthermore, the computational complexity of clustering has been extensively studied in computer science literature. Many of these results are negative, showing that clustering is computationally hard. For example, k-means clustering is NP-hard even for $k = 2$ [35], or in a 2-dimensional plane [83, 67]. In order to tackle the problem of computational complexity, some notions of niceness of data under which the clustering becomes easy have been considered (see [28] for a survey).

The closest proposal to the one that we will consider in this chapter is the notion of α-center proximity introduced by Awasthi et al. [16]. We discuss the relationship of these notions in Appendix 4.8. In the restricted scenario (i.e., when the centers of clusters are selected from the data set), their algorithm efficiently recovers the target clustering (outputs a tree such that the target is a pruning of the tree) for $\alpha > 3$. Balcan and Liang [21] improved the assumption to $\alpha > \sqrt{2} + 1$. Ben-David and Reyzin [29] showed that this problem is NP-Hard for $\alpha < 2$.

Variants of these proofs for our γ-margin condition yield the feasibility of k-means clustering when the input satisfies the condition with $\gamma > 2$ and NP hardness when $\gamma < 2$, both in the case of arbitrary (not necessarily Euclidean) metrics[2].

4.3 Problem Formulation

4.3.1 Center-based Clustering

The framework of clustering with queries can be applied to any type of clustering. However, in this work, we focus on a certain family of common clusterings – center-based clustering in Euclidean spaces[3].

Let $\mathcal{X}$ be a subset of some Euclidean space, $\mathbb{R}^d$. Let $\mathcal{C}_{\mathcal{X}} = \{C_1, \dots, C_k\}$ be a clustering (i.e., a partitioning) of $\mathcal{X}$. We say $x_1 \overset{C_{\mathcal{X}}}{\sim} x_2$ if x_1 and x_2 belong to the same cluster according to $C_{\mathcal{X}}$. We further denote by n the number of instances ($|\mathcal{X}|$) and by k the number of clusters.

We say that a clustering $C_{\mathcal{X}}$ is *center-based* if there exists a set of centers $\mu = \{\mu_1, \dots, \mu_k\} \subset \mathcal{R}^n$ such that the clustering corresponds to the Voroni diagram over those center points. Namely, for every x in $\mathcal{X}$ and $i \leq k$, $x \in C_i \Leftrightarrow i = \arg\min_j d(x, \mu_j)$.

Finally, we assume that the centers μ^* corresponding to C^* are the centers of mass of the corresponding clusters. In other words, $\mu_i^* = \frac{1}{|C_i|} \sum_{x \in C_i^*} x$. Note that this is the case for example when the oracle's clustering is the optimal solution to the Euclidean k-means clustering problem.

4.3.2 The γ-Margin Property

Next, we introduce a notion of clusterability of a data set, also referred to as 'data niceness property'.

Definition 4.1 (γ-margin)**.** *Let $\mathcal{X}$ be set of points in metric space M. Let $\mathcal{C}_{\mathcal{X}} = \{C_1, \dots, C_k\}$ be a center-based clustering of $\mathcal{X}$ induced by centers $\mu_1, \dots, \mu_k \in M$. We say that $\mathcal{C}_{\mathcal{X}}$ satisfies the γ-margin property if the following holds. For all $x \in C_i$ and $y \in C_j$,*

$$\gamma d(x, \mu_i) < d(y, \mu_i)$$

Similar notions have been considered before in the clustering literature. The closest one to our γ-margin is the notion of α-center proximity [21, 16]. We discuss the relationship between these two notions in Appendix 4.8.

[2] In particular, the hardness result of [29] relies on the ability to construct non-Euclidean distance functions. Later in this chapter, we prove hardness for $\gamma \leq \sqrt{3.4}$ for Euclidean instances.

[3] In fact, our results are all independent of the Euclidean dimension and apply to any Hilbert space.

4.3.3 The Algorithmic Setup

For a clustering $C^* = \{C_1^*, \ldots C_k^*\}$, a C^*-oracle is a function $\mathcal{O}_{C^*}$ that answers queries according to that clustering. One can think of such an oracle as a user that has some idea about its desired clustering, enough to answer the algorithm's queries. The clustering algorithm then tries to recover C^* by querying a C^*-oracle. The following notion of query is arguably most intuitive.

Definition 4.2 (Same-cluster Query)**.** *A same-cluster query asks whether two instances x_1 and x_2 belong to the same cluster, i.e.,*

$$\mathcal{O}_{C^*}(x_1, x_2) = \begin{cases} \textit{true} & \textit{if } x_1 \overset{C^*}{\sim} x_2 \\ \textit{false} & o.w. \end{cases}$$

(we omit the subscript C^* when it is clear from the context).

Definition 4.3 (Query Complexity)**.** *An SSAC instance is determined by the tuple $(\mathcal{X}, d, C^*)$. We will consider families of such instances determined by niceness conditions on their oracle clusterings C^*.*

1. *A SSAC algorithm $\mathcal{A}$ is called a q-solver for a family G of such instances, if for every instance in G, it can recover C^* by having access to $(\mathcal{X}, d)$ and making at most q queries to a C^*-oracle.*

2. *Such an algorithm is a polynomial q-solver if its time-complexity is polynomial in $|\mathcal{X}|$ and $|C^*|$ (the number of clusters).*

3. *We say G admits an $O(q)$ query complexity if there exists an algorithm $\mathcal{A}$ that is a polynomial q-solver for every clustering instance in G.*

4.4 An Efficient SSAC Algorithm

In this section we provide an efficient algorithm for clustering with queries. The setting is the one described in the previous section. In particular, it is assumed that the oracle has a center-based clustering in his mind which satisfies the γ-margin property. The space is Euclidean and the center of each cluster is the center of mass of the instances in that cluster. The algorithm not only makes same-cluster queries, but also another type of query defined as below.

Definition 4.4 (Cluster-assignment Query)**.** *A cluster-assignment query asks the cluster index that an instance x belongs to. In other words $\mathcal{O}_{C^*}(x) = i$ if and only if $x \in C_i^*$.*

Note however that each cluster-assignment query can be replaced with k same-cluster queries (see Appendix 4.7). Therefore, we can express everything in terms of the more natural notion of same-cluster queries, and the use of cluster-assignment query is just to make the representation of the algorithm simpler.

Intuitively, our proposed algorithm does the following. In the first phase, it tries to approximate the center of one of the clusters. It does this by asking cluster-assignment queries about a set of randomly (uniformly) selected point, until it has a sufficient number of points from at least one cluster (say C_p). It uses the mean of these points, μ'_p, to approximate the cluster center.

In the second phase, the algorithm recovers all of the instances belonging to C_p. In order to do that, it first sorts all of the instances based on their distance to μ'_p. By showing that all of the points in C_p lie inside a sphere centered at μ'_p (which does not include points from any other cluster), it tries to find the radius of this sphere by doing binary search using same-cluster queries. After that, the elements in C_p will be located and can be removed from the data set. The algorithm repeats this process k times to recover all of the clusters.

The details of our approach is stated precisely in Algorithm 1. Note that β is a small constant[4]. Theorem 4.1 shows that if $\gamma > 1$ then our algorithm recovers the target clustering with high probability. Next, we give bounds on the time and query complexity of our algorithm. Theorem 4.2 shows that our approach needs $O(k \log n + k^2 \log k)$ queries and runs with time complexity $O(kn \log n)$.

Lemma 4.1. *Let $(\mathcal{X}, d, C)$ be a clustering instance, where C is center-based and satisfies the γ-margin property. Let μ be the set of centers corresponding to the centers of mass of C. Let μ'_i be such that $d(\mu_i, \mu'_i) \leq r(C_i)\epsilon$, where $r(C_i) = \max_{x \in C_i} d(x, \mu_i)$. Then $\gamma \geq 1 + 2\epsilon$ implies that*

$$\forall x \in C_i, \forall y \in \mathcal{X} \setminus C_i \Rightarrow d(x, \mu'_i) < d(y, \mu'_i)$$

Proof. Fix any $x \in C_i$ and $y \in C_j$. $d(x, \mu'_i) \leq d(x, \mu_i) + d(\mu_i, \mu'_i) \leq r(C_i)(1 + \epsilon)$. Similarly, $d(y, \mu'_i) \geq d(y, \mu_i) - d(\mu_i, \mu'_i) > (\gamma - \epsilon)r(C_i)$. Combining the two, we get that $d(x, \mu'_i) < \frac{1+\epsilon}{\gamma-\epsilon} d(y, \mu'_i)$. □

Lemma 4.2. *Let the framework be as in Lemma 4.1. Let Z_p, C_p, μ_p, μ'_p and η be defined as in Algorhtm 1, and $\epsilon = \frac{\gamma-1}{2}$. If $|Z_p| > \eta$, then the probability that $d(\mu_p, \mu'_p) > r(C_p)\epsilon$ is at most $\frac{\delta}{k}$.*

Proof. Define a uniform distribution U over C_p. Then μ_p and μ'_p are the true and empirical mean of this distribution. Using a standard concentration inequality (Theorem 4.9 from Appendix 4.10) shows that the empirical mean is close to the true mean, completing the proof.

[4] It corresponds to the constant appeared in generalized Hoeffding inequality bound, discussed in Theorem 4.9 in appendix 4.10 in supplementary materials.

Algorithm 1: Algorithm for $\gamma(> 1)$-margin instances with queries

```
Input: Clustering instance $\mathcal{X}$, oracle $\mathcal{O}$, the number of clusters $k$ and parameter $\delta \in (0,1)$
Output: A clustering $\mathcal{C}$ of the set $\mathcal{X}$
$\mathcal{C} = \{\}, \mathcal{S}_1 = \mathcal{X}, \eta = \beta\frac{\log k + \log(1/\delta)}{(\gamma-1)^4}$
for $i = 1$ to $k$ do
    Phase 1
    $l = k\eta + 1$;
    $Z \sim U^l[\mathcal{S}_i]$  // Draws $l$ independent elements from $\mathcal{S}_i$ uniformly at random
    For $1 \leq t \leq i$,
      $Z_t = \{x \in Z : \mathcal{O}(x) = t\}$.  //Asks cluster-assignment queries about the members of
      $Z$
    $p = \arg\max_t |Z_t|$
    $\mu'_p := \frac{1}{|Z_p|}\sum_{x \in Z_p} x$.
    Phase 2
    // We know that there exists $r_i$ such that $\forall x \in \mathcal{S}_i, x \in C_i \Leftrightarrow d(x, \mu'_i) < r_i$.
    // Therefore, $r_i$ can be found by simple binary search
    $\widehat{\mathcal{S}_i}$ = Sorted($\{\mathcal{S}_i\}$)  // Sorts elements of $\{x : x \in \mathcal{S}_i\}$ in increasing order of $d(x, \mu'_p)$.
    $r_i$ = BinarySearch($\widehat{\mathcal{S}_i}$)  //This step takes up to $O(\log|\mathcal{S}_i|)$ same-cluster queries
    $C'_p = \{x \in \mathcal{S}_i : d(x, \mu'_p) \leq r_i\}$.
    $S_{i+1} = S_i \setminus C'_p$.
    $\mathcal{C} = \mathcal{C} \cup \{C'_p\}$
end
```

□

Theorem 4.1. *Let $(\mathcal{X}, d, C)$ be a clustering instance, where C is center-based and satisfies the γ-margin property. Let μ_i be the center corresponding to the center of mass of C_i. Assume $\delta \in (0,1)$ and $\gamma > 1$. Then with probability at least $1-\delta$, Algorithm 1 outputs C.*

Proof. In the first phase of the algorithm we are making $l > k\eta$ cluster-assignment queries. Therefore, using the pigeonhole principle, we know that there exists cluster index p such that $|Z_p| > \eta$. Then Lemma 4.2 implies that the algorithm chooses a center μ'_p such that with probability at least $1 - \frac{\delta}{k}$ we have $d(\mu_p, \mu'_p) \leq r(C_p)\epsilon$. By Lemma 4.1, this would mean that $d(x, \mu'_p) < d(y, \mu'_p)$ for all $x \in C_p$ and $y \notin C_p$. Hence, the radius r_i found in the phase two of Alg. 1 is such that $r_i = \max_{x \in C_p} d(x, \mu'_p)$. This implies that C'_p (found in phase two) equals to C_p.

Hence, with probability at least $1 - \frac{\delta}{k}$ one iteration of the algorithm successfully finds all the points in a cluster C_p. Using union bound, we get that with probability at least $1 - k\frac{\delta}{k} = 1 - \delta$, the algorithm recovers the target clustering. □

Theorem 4.2. *Let the framework be as in theorem 4.1. Then Algorithm 1*

- *Makes* $O\big(k \log n + k^2 \frac{\log k + \log(1/\delta)}{(\gamma-1)^4}\big)$ *same-cluster queries to the oracle* $\mathcal{O}$.
- *Runs in* $O\big(kn \log n + k^2 \frac{\log k + \log(1/\delta)}{(\gamma-1)^4}\big)$ *time.*

Proof. In each iteration (i) the first phase of the algorithm takes $O(\eta)$ time and makes $\eta + 1$ cluster-assignment queries (ii) the second phase takes $O(n \log n)$ times and makes $O(\log n)$ same-cluster queries. Each cluster-assignment query can be replaced with k same-cluster queries; therefore, each iteration runs in $O(k\eta + n \log n)$ and uses $O(k\eta + \log n)$ same-cluster queries. By replacing $\eta = \beta \frac{\log k + \log(1/\delta)}{(\gamma-1)^4}$ and noting that there are k iterations, the proof will be complete. □

Corollary 4.1. *The set of Euclidean clustering instances that satisfy the* γ*-margin property for some* $\gamma > 1$ *admits query complexity* $O\big(k \log n + k^2 \frac{\log k + \log(1/\delta)}{(\gamma-1)^4}\big)$.

Remark 4.1. *In fact, the algorithm does not need to know the value of k in advance: a new cluster should be created whenever the queried instance does not match with any of the (representatives of the) current clusters (through same-cluster queries).*

4.5 Hardness Results

4.5.1 Hardness of Euclidean k-Means with Margin

Finding k-means solution without the help of an oracle is generally computationally hard. In this section, we will show that solving Euclidean k-means remains hard even if we know that the optimal solution satisfies the γ-margin property for $\gamma = \sqrt{3.4}$. In particular, we show the hardness for the case of $k = \Theta(n^\epsilon)$ for any $\epsilon \in (0, 1)$.

In Section 4.4, we proposed a polynomial-time algorithm that could recover the target clustering using $O(k^2 \log k + k \log n)$ queries, assuming that the clustering satisfies the γ-margin property for $\gamma > 1$. Now assume that the oracle conforms to the optimal k-means clustering solution. In this case, for $1 < \gamma \leq \sqrt{3.4} \approx 1.84$, solving k-means clustering would be NP-hard without queries, while it becomes efficiently solvable with the help of an oracle [5].

[5] To be precise, note that the algorithm used for clustering with queries is probabilistic, while the lower bound that we provide is for deterministic algorithms. However, this implies a lower bound for randomized algorithms as well unless $BPP \neq P$

Given a set of instances $\mathcal{X} \subset \mathbf{R}^d$, the k-means clustering problem is to find a clustering $\mathcal{C} = \{C_1, \ldots, C_k\}$ which minimizes $f(\mathcal{C}) = \sum\limits_{C_i} \min\limits_{\mu_i \in \mathbf{R}^d} \sum\limits_{x \in C_i} \|x - \mu_i\|_2^2$. The decision version of k-means is, given some value L, is there a clustering $\mathcal{C}$ with cost $\leq L$? The following theorem is the main result of this section.

Theorem 4.3. *Finding the optimal solution to Euclidean k-means objective function is NP-hard when $k = \Theta(n^\epsilon)$ for any $\epsilon \in (0, 1)$, even when the optimal solution satisfies the γ-margin property for $\gamma = \sqrt{3.4}$.*

This results extends the hardness result of [29] to the case of Euclidean metric, rather than arbitrary one, and to the γ-margin condition (instead of the α-center proximity there). The full proof is rather technical and is deferred to the Appendix 4.9. In the next sections, we provide an outline of the proof.

4.5.2 Overview of the Proof

Our method to prove Theorem 4.3 is based on the approach employed by [83]. However, the original construction proposed in [83] does not satisfy the γ-margin property. Therefore, we have to modify the proof by setting up the parameters of the construction more carefully.

To prove the theorem, we will provide a reduction from the problem of Exact Cover by 3-Sets (X3C) which is NP-Complete [50], to the decision version of k-means.

Definition 4.5 (X3C)**.** *Given a set U containing exactly $3m$ elements and a collection $\mathcal{S} = \{S_1, \ldots, S_l\}$ of subsets of U such that each S_i contains exactly three elements, does there exist m elements in $\mathcal{S}$ such that their union is U?*

We will show how to translate each instance of X3C, $(U, \mathcal{S})$, to an instance of k-means clustering in the Euclidean plane, X. In particular, X has a grid-like structure consisting of l rows (one for each S_i) and roughly $6m$ columns (corresponding to U) which are embedded in the Euclidean plane. The special geometry of the embedding makes sure that any low-cost k-means clustering of the points (where k is roughly $6ml$) exhibits a certain structure. In particular, any low-cost k-means clustering could cluster each row in only two ways; One of these corresponds to S_i being included in the cover, while the other means it should be excluded. We will then show that U has a cover of size m if and only if X has a clustering of cost less than a specific value L. Furthermore, our choice of embedding makes sure that the optimal clustering satisfies the γ-margin property for $\gamma = \sqrt{3.4} \approx 1.84$.

4.5.3 Reduction Design

Given an instance of X3C, that is the elements $U = \{1, \dots, 3m\}$ and the collection $\mathcal{S}$, we construct a set of points X in the Euclidean plane which we want to cluster. Particularly, X consists of a set of points $H_{l,m}$ in a grid-like manner, and the sets Z_i corresponding to S_i. In other words, $X = H_{l,m} \cup (\cup_{i=1}^{l-1} Z_i)$.

The set $H_{l,m}$ is as described in Fig. 4.1. The row R_i is composed of $6m + 3$ points $\{s_i, r_{i,1}, \dots, r_{i,6m+1}, f_i\}$. Row G_i is composed of $3m$ points $\{g_{i,1}, \dots, g_{i,3m}\}$. The distances between the points are also shown in Fig. 4.1. Also, all these points have weight w, simply meaning that each point is actually a set of w points on the same location.

Each set Z_i is constructed based on S_i. In particular, $Z_i = \cup_{j\in[3m]} B_{i,j}$, where $B_{i,j}$ is a subset of $\{x_{i,j}, x'_{i,j}, y_{i,j}, y'_{i,j}\}$ and is constructed as follows: $x_{i,j} \in B_{i,j}$ *iff* $j \notin S_i$, and $x'_{i,j} \in B_{i,j}$ *iff* $j \in S_i$. Similarly, $y_{i,j} \in B_{i,j}$ *iff* $j \notin S_{i+1}$, and $y'_{i,j} \in B_{i,j}$ *iff* $j \in S_{i+1}$. Furthermore, $x_{i,j}, x'_{i,j}, y_{i,j}$ and $y'_{i,j}$ are specific locations as depicted in Fig. 4.2. In other words, exactly one of the locations $x_{i,j}$ and $x'_{i,j}$, and one of $y_{i,j}$ and $y'_{i,j}$ will be occupied. We set the following parameters.

$$h = \sqrt{5}, d = \sqrt{6}, \epsilon = \frac{1}{w^2}, \lambda = \frac{2}{\sqrt{3}}h, k = (l-1)3m + l(3m+2)$$

$$L_1 = (6m+3)wl, L_2 = 3m(l-1)w, L = L_1 + L_2 - m\alpha, \alpha = \frac{d}{w} - \frac{1}{2w^3}$$

Lemma 4.3. *The set $X = H_{l,n} \cup Z$ has a k-clustering of cost less or equal to L if and only if there is an exact cover for the X3C instance.*

Lemma 4.4. *Any k-clustering of $X = H_{l,n} \cup Z$ with cost $\leq L$ has the γ-margin property where $\gamma = \sqrt{3.4}$. Furthermore, $k = \Theta(n^\epsilon)$.*

The proofs are provided in Appendix 4.9. Lemmas 4.3 and 4.4 together show that X has a k-clustering of cost $\leq L$ satisfying the γ-margin property (for $\gamma = \sqrt{3.4}$) if and only if there is an exact cover by 3-sets for the X3C instance. This completes the proof of Theorem 4.3.

4.5.4 Lower Bound on the Number of Queries

In the previous section we showed that k-means clustering is NP-hard even under γ-margin assumption (for $\gamma < \sqrt{3.4} \approx 1.84$). On the other hand, in Section 4.4 we showed that this is not the case if the algorithm has access to an oracle. In this section, we show a lower bound on the number of queries needed to provide a polynomial-time algorithm for k-means clustering under margin assumption.

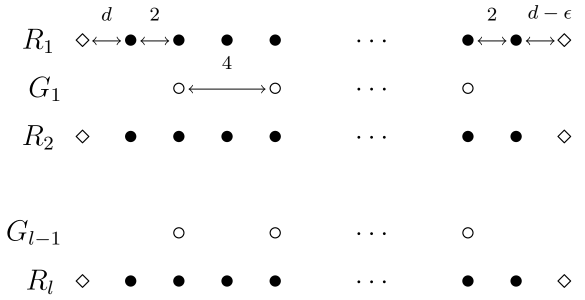

Figure 4.1: Geometry of $H_{l,m}$. This figure is similar to Fig. 1 in [83]. Reading from left to right, each row R_i consists of a diamond (s_i), $6m+1$ bullets ($r_{i,1}, \ldots, r_{i,6m+1}$), and another diamond ($f_i$). Each rows G_i consists of $3m$ circles ($g_{i,1}, \ldots, g_{i,3m}$).

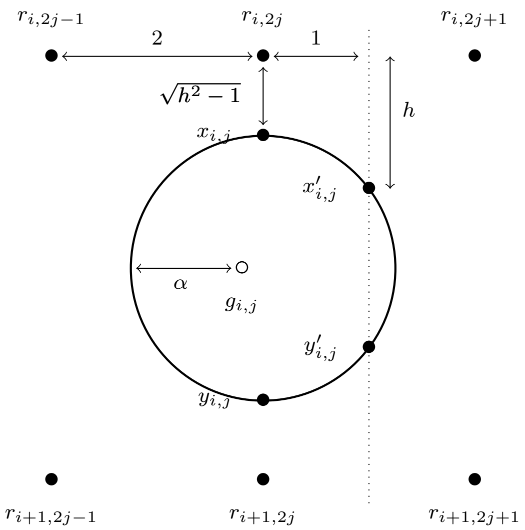

Figure 4.2: The locations of $x_{i,j}$, $x'_{i,j}$, $y_{i,j}$ and $y'_{i,j}$ in the set Z_i. Note that the point $g_{i,j}$ is not vertically aligned with $x_{i,j}$ or $r_{i,2j}$. This figure is adapted from [83].

Theorem 4.4. *For any $\gamma \leq \sqrt{3.4}$, finding the optimal solution to the k-means objective function is NP-Hard even when the optimal clustering satisfies the γ-margin property and the algorithm can ask $O(\log k + \log |\mathcal{X}|)$ same-cluster queries.*

Proof. Proof by contradiction: assume that there is a polynomial-time algorithm $\mathcal{A}$ that makes $O(\log k + \log |\mathcal{X}|)$ same-cluster queries to the oracle. Then, we show there exists another algorithm $\mathcal{A}'$ for the same problem that is still polynomial but uses no queries. However, this will be a contradiction to Theorem 4.3, which will prove the result.

In order to prove that such $\mathcal{A}'$ exists, we use a 'simulation' technique. Note that $\mathcal{A}$ makes only $q < \beta(\log k + \log |\mathcal{X}|)$ binary queries, where β is a constant. The oracle therefore can respond to these queries in maximum $2^q < k^\beta |\mathcal{X}|^\beta$ different ways. Now the algorithm $\mathcal{A}'$ can try to simulate all of $k^\beta |\mathcal{X}|^\beta$ possible responses by the oracle and output the solution with minimum k-means clustering cost. Therefore, $\mathcal{A}'$ runs in polynomial-time and is equivalent to $\mathcal{A}$. □

4.6 Conclusions

In this chapter we introduced a framework for semi-supervised active clustering (SSAC) with same-cluster queries. Those queries can be viewed as a natural way for a clustering mechanism to gain domain knowledge, without which clustering is an under-defined task. The focus of our analysis was the computational and query complexity of such SSAC problems, when the input data set satisfies a clusterability condition – the γ-margin property.

Our main result shows that access to a limited number of such query answers (logarithmic in the size of the data set and quadratic in the number of clusters) allows efficient successful clustering under conditions (margin parameter between 1 and $\sqrt{3.4} \approx 1.84$) that render the problem NP-hard without the help of such a query mechanism. We also provided a lower bound indicating that at least $\Omega(\log kn)$ queries are needed to make those NP hard problems feasibly solvable.

With practical applications of clustering in mind, a natural extension of this model is to allow the oracle (i.e., the domain expert) to refrain from answering a certain fraction of the queries, or to make a certain number of errors in its answers. It would be interesting to analyze how the performance guarantees of SSAC algorithms behave as a function of such abstentions and error rates. Interestingly, we can modify our algorithm to handle a sub-logarithmic number of abstentions by checking all possible oracle answers to them (i.e., similar to the "simulation" trick in the proof of Theorem 4.4).

4.6.1 Subsequent Results

In this section we will mention some of the recent subsequent results that have been shown by other researchers about the semi-supervised active clustering (SSAC) model.

Recall that the algorithm that we proposed recovers the *exact* optimal clustering (with high probability), assuming that the solution satisfies some niceness (i.e., γ-margin) condition. Ailon et al. [4] removed the niceness assumption, providing an efficient *approximate* solution (in terms of k-means cost) to the k-means clustering problem using queries. Again, this result demonstrates that the computationally hard problem of approximate clustering can be solved efficiently if the learner has access to same-cluster queries. More recently, Gamlath et al. [49] extended this analysis to the case where we are not only interested in minimizing the k-means loss, but we would also want to recover (approximately) the original partition of the instances.

The SSAC model has been adapted for other clustering algorithms as well. Ailon et al. [3] used same-cluster queries to solve the approximate correlation clustering problem. Also, Mazumdar and Saha [72] considered clustering in the stochastic block model using same-cluster queries.

The SSAC framework has been extended to the case of noisy oracle as well. Kim and Ghosh [55, 56] considered "weak" oracles that can convey a degree of confidence (based on the geometry of the points) for the query answers. Mazumdar and Saha [71] studied the crowd sourcing scenario where the teachers (i.e., oracles) output the correct answer with some fixed probability p (they consider various clustering schemes, including correlation clustering and stochastic block model). Finally, the relationship between SSAC model and *locally encodable source coding* was investigated by Mazumdar and Saha [70], showing some information-theoretic limitations of same-cluster queries in the unstructured setting (e.g., when the geometry of the instances is unavailable), and offering the use of *AND queries* instead.

4.7 Appendix: Relationships Between Query Models

Proposition 4.1. *Any clustering algorithm that uses only q same-cluster queries can be adjusted to use $2q$ cluster-assignment queries (and no same-cluster queries) with the same order of time complexity.*

Proof. We can replace each same-cluster query with two cluster-assignment queries as in $Q(x_1, x_2) = \mathbb{1}\{Q(x_1) = Q(x_2))\}$. □

Proposition 4.2. *Any algorithm that uses only q cluster-assignment queries can be adjusted to use kq same-cluster queries (and no cluster-assignment queries) with at most a factor k increase in computational complexity, where k is the number of clusters.*

Proof. If the clustering algorithm has access to an instance from each of k clusters (say $x_i \in X_i$), then it can simply simulate the cluster-assignment query by making k same-cluster queries ($Q(x) = \arg\max_i \mathbb{1}\{Q(x, x_i)\}$). Otherwise, assume that at the time of querying $Q(x)$ it has only instances from $k' < k$ clusters. In this case, the algorithm can do the same with the k' instances and if it does not find the cluster, assign x to a new cluster index. This will work, because in the clustering task the output of the algorithm is a partition of the elements, and therefore the indices of the clusters do not matter. □

4.8 Appendix: Comparison of γ-Margin and α-Center Proximity

In this chapter, we introduced the notion of γ-margin niceness property. We further showed upper and lower bounds on the computational complexity of clustering under this assumption. It is

Table 4.1: Known results for α-center proximity

	Euclidean	General Metric
Centers from data	Upper bound : $\sqrt{2}+1$ [21] Lower bound : ?	Upper bound : $\sqrt{2}+1$ [21] Lower bound : 2 [29]
Unrestricted Centers	Upper bound : $2+\sqrt{3}$ [16] Lower bound : ?	Upper bound : $2+\sqrt{3}$ [16] Lower bound : 3 [16]

therefore important to compare this notion with other previously-studied clusterability notions.

An important notion of niceness of data for clustering is α-center proximity property.

Definition 4.6 (α-center proximity [16]). *Let $(\mathcal{X}, d)$ be a clustering instance in some metric space M, and let k be the number of clusters. We say that a center-based clustering $\mathcal{C}_{\mathcal{X}} = \{C_1, \ldots, C_k\}$ induced by centers $c_1, \ldots, c_k \in M$ satisfies the α-center proximity property (with respect to $\mathcal{X}$ and k) if the following holds*

$$\forall x \in C_i, i \neq j, \alpha d(x, c_i) < d(x, c_j)$$

This property has been considered in the past in various studies [21, 16]. In this appendix we will show some connections between the notions of γ-margin and α-center proximity.

It is important to note that throughout this chapter we considered clustering in Euclidean spaces. Furthermore, the centers were not restricted to be selected from the data points. However, this is not necessarily the case in other studies.

An overview of the known results under α-center proximity is provided in Table 4.1. The results are provided for the case that the centers are restricted to be selected from the training set, and also the unrestricted case (where the centers can be arbitrary points from the metric space). Note that any upper bound that works for general metric spaces also works for the Euclidean space.

We will show that using the same techniques one can prove upper and lower bounds for γ-margin property. It is important to note that for γ-margin property, in some cases the upper and lower bounds match. Hence, there is no hope to further improve those bounds unless P=NP. A summary of our results is provided in 4.2.

4.8.1 Centers from Input Instances

Theorem 4.5. *Let (X, d) be a clustering instance and $\gamma \geq 2$. Then, Algorithm 1 in [21] outputs a tree $\mathcal{T}$ with the following property:*

Table 4.2: Results for γ-margin

	Euclidean	General Metric
Centers from data	Upper bound : 2 (Thm. 4.5) Lower bound : ?	Upper bound : 2 (Thm. 4.5) Lower bound : 2 (Thm. 4.6)
Unrestricted Centers	Upper bound : 3 (Thm. 4.7) Lower bound : 1.84 (Thm. 4.3)	Upper bound : 3 (Thm. 4.7) Lower bound : 3 (Thm. 4.8) Awasthi

Any k-clustering $\mathcal{C}^* = \{C_1^*, \ldots, C_k^*\}$ which satisfies the γ-margin property and its cluster centers $\mu_1, \ldots, \mu_k$ are in X, is a pruning of the tree T. In other words, for every $1 \leq i \leq k$, there exists a node N_i in the tree T such that $C_i^* = N_i$.

Proof. Let $p, p' \in C_i^*$ and $q \in C_j^*$. [21] prove the correctness of their algorithm for $\alpha > \sqrt{2} + 1$. Their proof relies only on the following three properties which are implied when $\alpha > \sqrt{2} + 1$. We will show that these properties are implied by $\gamma > 2$ instances as well.

- $d(p, \mu_i) < d(p, q)$
  $\gamma d(p, \mu_i) < d(q, \mu_i) < d(p, q) + d(p, \mu_i) \implies d(p, \mu_i) < \frac{1}{\gamma - 1} d(p, q)$.
- $d(p, \mu_i) < d(q, \mu_i)$
  This is trivially true since $\gamma > 2$.
- $d(p, \mu_i) < d(p', q)$
  Let $r = \max_{x \in C_i^*} d(x, \mu_i)$. Observe that $d(p, \mu_i) < r$. Also, $d(p', q) > d(q, \mu_i) - d(p', \mu_i) > \gamma r - r = (\gamma - 1) r$.

□

Theorem 4.6. *Let $(\mathcal{X}, d)$ be a clustering instance and k be the number of clusters. For $\gamma < 2$, finding a k-clustering of X which satisfies the γ-margin property and where the corresponding centers $\mu_1, \ldots, \mu_k$ belong to $\mathcal{X}$ is NP-Hard.*

Proof. For $\alpha < 2$, [29] proved that in general metric spaces, finding a clustering which satisfies the α-center proximity and where the centers $\mu_1, \ldots, \mu_k \in \mathcal{X}$ is NP-Hard. Note that the reduced instance in their proof, also satisfies γ-margin for $\gamma < 2$. □

4.8.2 Unrestricted Centers from the Metric Space

Theorem 4.7. *Let (X, d) be a clustering instance and $\gamma \geq 3$. Then, the standard single-linkage algorithm outputs a tree $\mathcal{T}$ with the following property:*

Any k-clustering $\mathcal{C}^* = \{C_1^*, \ldots, C_k^*\}$ which satisfies the γ-margin property is a pruning of T. In other words, for every $1 \leq i \leq k$, there exists a node N_i in the tree T such that $C_i^* = N_i$.

Proof. [20] showed that if a clustering C^* has the strong stability property, then single-linkage outputs a tree with the required property. It is simple to see that if $\gamma > 3$ then instances have strong-stability and the claim follows. □

Theorem 4.8. *Let $(\mathcal{X}, d)$ be a clustering instance and $\gamma < 3$. Then, finding a k-clustering of X which satisfies the γ-margin is NP-Hard.*

Proof. [16] proved the above claim but for $\alpha < 3$ instances. Note however that the construction in their proof satisfies γ-margin for $\gamma < 3$. □

4.9 Appendix: Proofs of Lemmas 4.3 and 4.4

In Section 4.5 we proved Theorem 4.3 based on two technical results (i.e., lemma 4.3 and 4.4). In this appendix we provide the proofs for these lemmas. In order to start, we first need to establish some properties about the Euclidean embedding of X proposed in Section 4.5.

Definition 4.7 (A- and B-Clustering of R_i). *An A-Clustering of row R_i is a clustering in the form of $\{\{s_i\}, \{r_{i,1}, r_{i,2}\}, \{r_{i,3}, r_{i,4}\}, \ldots, \{r_{i,6m-1}, r_{i,6m}\}, \{r_{i,6m+1}, f_i\}\}$. A B-Clustering of row R_i is a clustering in the form of $\{\{s_i, r_{i,1}\}, \{r_{i,2}, r_{i,3}\}, \{r_{i,4}, r_{i,5}\}, \ldots, \{r_{i,6m}, r_{i,6m+1}\}, \{f_i\}\}$.*

Definition 4.8 (Good point for a cluster). *A cluster C is good for a point $z \notin C$ if adding z to C increases cost by exactly $\frac{2w}{3}h^2$.*

Given the above definition, the following simple observations can be made.

- The clusters $\{r_{i,2j-1}, r_{i,2j}\}$, $\{r_{i,2j}, r_{i,2j+1}\}$ and $\{g_{i,j}\}$ are good for $x_{i,j}$ and $y_{i-1,j}$.
- The clusters $\{r_{i,2j}, r_{i,2j+1}\}$ and $\{g_{i,j}\}$ are good for $x'_{i,j}$ and $y'_{i-1,j}$.

Definition 4.9 (Nice Clustering). *A k-clusteirng is nice if every $g_{i,j}$ is a singleton cluster, each R_i is grouped in the form of either an A-clustering or a B-clustering, and each point in Z_i is added to a cluster which is good for it.*

It is straightforward to see that a row grouped in an A-clustering costs $(6m+3)w - \alpha$ while a row in B-clustering costs $(6m+3)w$. Hence, a nice clustering of $H_{l,m} \cup Z$ costs at most $L_1 + L_2$. More specifically, if t rows are grouped in an A-clustering, the nice-clustering costs $L_1 + L_2 - t\alpha$. Also, observe that any nice clustering of X has only the following four different types of clusters.

(1) Type E - $\{r_{i,2j-1}, r_{i,2j+1}\}$
The cost of this cluster is $2w$ and the contribution of each location to the cost (i.e., $\frac{cost}{\#locations}$) is $\frac{2w}{2} = w$.

(2) Type F - $\{r_{i,2j-1}, r_{i,2j}, x_{i,j}\}$ or $\{r_{i,2j-1}, r_{i,2j}, y_{i-1,j}\}$ or $\{r_{i,2j}, r_{i,2j+1}, x'_{i,j}\}$ or $\{r_{i,2j}, r_{i,2j+1}, y'_{i-1,j}\}$
The cost of any cluster of this type is $2w(1 + \frac{h^2}{3})$ and the contribution of each location to the cost is at most $\frac{2w}{9}(h^2+3)$. This is equal to $\frac{16}{9}w$ because we had set $h = \sqrt{5}$.

(3) Type I - $\{g_{i,j}, x_{i,j}\}$ or $\{g_{i,j}, x'_{i,j}\}$ or $\{g_{i,j}, y_{i,j}\}$ or $\{g_{i,j}, y'_{i,j}\}$
The cost of any cluster of this type is $\frac{2}{3}wh^2$ and the contribution to the cost of each location is $\frac{w}{3}h^2$. For our choice of h, the contribution is $\frac{5}{3}w$.

(4) Type J - $\{s_i, r_{i,1}\}$ or $\{r_{i,6m+1}, f_i\}$
The cost of this cluster is $3w$ (or $3w - \alpha$) and the contribution of each location to the cost is at most $1.5w$.

Hence, observe that in a nice-clustering, any location contributes at most $\leq \frac{16}{9}w$ to the total clustering cost. This observation will be useful in the proof of the lemma below.

Lemma 4.5. *For large enough $w = poly(l, m)$, any non-nice clustering of $X = H_{l,m} \cup Z$ costs at least $L + \frac{w}{3}$.*

Proof. We will show that any non-nice clustering C of X costs at least $\frac{w}{3}$ more than any nice clustering. This will prove our result. The following cases are possible.

- C contains a cluster C_i of cardinality $t > 6$ (i.e., contains t weighted points)
Observe that any $x \in C_i$ has at least $t-5$ locations at a distance greater than 4 to it, and 4 locations at a distance at least 2 to it. Hence, the cost of C_i is at least $\frac{w}{2t}(4^2(t-5) + 2^2 4)t = 8w(t-4)$. C_i allows us to use at most $t-2$ singletons. This is because a nice clustering of these $t + (t-2)$ points uses at most $t-1$ clusters and the clustering C uses $1 + (t-2)$ clusters for these points. The cost of the nice cluster on these points is $\leq \frac{16w}{9}2(t-1)$. While the non-nice clustering costs at least $8w(t-4)$. For $t \geq 6.4 \implies 8(t-4) > \frac{32}{9}(t-1)$ and the claim follows. Note that in this case the difference in cost is at least $\frac{8w}{3}$.
- C contains a cluster of cardinality $t = 6$
Note that among all clusters of cardinality 6, the following has the minimum cost: $C_i = \{r_{i,2j-1}, r_{i,2j}, x_{i,j}, y_{i-1,j}, r_{i,2j+1}, r_{2j+2}\}$. The cost of this cluster is $\frac{176w}{6}$. Arguing as before, this allows us to use 4 singletons. Hence, a nice cluster on these 10 points costs at most $\frac{160w}{9}$. The difference of cost is at least $34w$.

- C contains a cluster of cardinality $t = 5$
  Note that among all clusters of cardinality 5, the following has the minimum cost: $C_i = \{r_{i,2j-1}, r_{i,2j}, x_{i,j}, y_{i-1,j}, r_{i,2j+1}\}$. The cost of this cluster is $16w$. Arguing as before, this allows us to use 3 singletons. Hence, a nice cluster on these 8 points costs at most $16w\frac{8}{9}$. The difference of cost is at least $\frac{16w}{9}$.
- C contains a cluster of cardinality $t = 4$
  It is easy to see that amongst all clusters of cardinality 4, the following has the minimum cost. $C_i = \{r_{i,2j-1}, r_{i,2j}, x_{i,j}, r_{i,2j+1}\}$. The cost of this cluster is $11w$. Arguing as before, this allows us to use 2 singletons. Hence, a nice cluster on these 6 points costs at most $\frac{32w}{3}$. The difference of cost is at least $\frac{w}{3}$.
- All the clusters have cardinality ≤ 3
  Observe that amongst all non-nice clusters of cardinality 3, the following has the minimum cost: $C_i = \{r_{i,2j-1}, r_{i,2j}, r_{i,2j+1}\}$. The cost of this cluster is $8w$. Arguing as before, this allows us to use at most 1 more singleton. Hence, a nice cluster on these 4 points costs at most $\frac{64w}{9}$. The difference of cost is at least $\frac{8w}{9}$.
  It is also simple to see that any non-nice clustering of size 2 increases the cost by at least w.

□

Proof of lemma 4.3. The proof is identical to the proof of Lemma 11 in [83]. Note that the parameters that we use are different with those utilized by [83]; however, this is not an issue, because we can invoke our lemma 4.5 instead of the analogous result in Vattani (i.e., lemma 10 in Vattani's paper). The sketch of the proof is that based on lemma 4.5, only nice clusterings of X cost $\leq L$. On the other hand, a nice clustering corresponds to an exact 3-set cover. Therefore, if there exists a clustering of X of cost $\leq L$, then there is an exact 3-set cover. The other way is simpler to proof; assume that there exists an exact 3-set cover. Then, the corresponding construction of X makes sure that it will be clustered *nicely*, and therefore will cost $\leq L$.

□

Proof of lemma 4.4. As argued before, any nice clustering has four different types of clusters. We will calculate the minimum ratio $a_i = \frac{d(y,\mu)}{d(x,\mu)}$ for each of these clusters C_i (where $x \in C_i$, $y \notin C_i$ and μ is mean of all the points in C_i.) Then, the minimum a_i will give the desired γ.

(1) For Type E clusters $a_i = h/1 = \sqrt{5}$.

(2) For Type F clusters. $a_i = \frac{\frac{\sqrt{4+16(h^2-1)}}{3}}{2h/3} = \sqrt{\frac{17}{5}} \approx 1.84$.

(3) For Type I clusters, standard calculation show that $a_i > 2$.

(4) For Type J clusters $a_i = \frac{2+\frac{\sqrt{6}}{2}}{\frac{\sqrt{6}}{2}} > 2$.

Furthermore, $|\mathcal{X}| = (12lm + 3l - 6m)w$ and $k = 6lm + 2l - 3m$. Hence for w =poly(l, m) our hardness result holds for $k = |\mathcal{X}|^\epsilon$ for any $0 < \epsilon < 1$. □

Lemmas 4.3 and 4.4 complete the proof of the main result (Theorem 4.3).

4.10 Appendix: Concentration Inequality

Theorem 4.9 (Generalized Hoeffding's Inequality (e.g., [13]))**.** *Let $X_1, \ldots .X_n$ be i.i.d random vectors in some Hilbert space such that for all i, $\|X_i\|_2 \leq R$ and $E[X_i] = \mu$. If $n > c\frac{\log(1/\delta)}{\epsilon^2}$, then with probability at least $1 - \delta$, we have that*

$$\left\|\mu - \frac{1}{n}\sum X_i\right\|_2^2 \leq R^2\epsilon$$

Chapter 5

Learning Mixture Models: Background

Learning distributions is a fundamental problem in statistics and computer science, and has numerous applications in machine learning and signal processing. The problem can be stated as:

> Given an i.i.d. sample generated from an unknown probability distribution g, find a distribution $\hat{g}$ that is close to g in total variation distance.[1]

This strong notion of learning is not possible in general using a finite number of samples. However, if we assume that the target distribution belongs to or can be approximated by a family of distributions, then there is hope to acquire algorithms with finite-sample guarantees. In this chapter, we study learning the important families of mixture models within this framework. As an example of this setting, assume that the target distribution is a Gaussian mixture with k components in $\mathbb{R}^d$. Then, how many examples do we need to find a distribution that is ε-close to the target? This *sample complexity* question, as well as the corresponding *computational complexity* question, has received a lot of attention recently (see, e.g. [48, 33, 79, 41, 46, 1]).

We also want to study a related setting, where the learner can ask queries about the actual mixture component that an instance is generated from. We will then investigate if these queries can help reducing the computational or statistical complexity of learning mixture models. This new direction can be thought as generalizing the results of the previous chapter about center-based clustering to the mixture learning setting.

Notice that we consider PAC learning of distributions (a.k.a. density estimation), which is different from parameter estimation. In the parameter estimation problem, it is assumed that

[1] Total variation distance is a prominent distance measure between distributions. For a discussion on this and other choices see [39, Chapter 5].

the target distribution belongs to some parametric class, and the goal is to learn/identify the parameters (see, e.g., [34, 26, 76]).

5.1 The Formal Framework

Generally speaking, a *distribution learning method* is an algorithm that takes a sample of i.i.d. points from distribution g as input, and outputs (a description) of a distribution $\hat{g}$ as an estimation for g. Furthermore, we assume that g belongs to or can be approximated by class $\mathcal{F}$ of distributions, and we may require that $\hat{g}$ also belongs to this class (i.e., proper learning).

Let f_1 and f_2 be two probability distributions defined over the Borel σ-algebra $\mathcal{B}$. The total variation distance between f_1 and f_2 is defined as

$$\|f_1 - f_2\|_{TV} = \sup_{B \in \mathcal{B}} |f_1(B) - f_2(B)| = \frac{1}{2}\|f_1 - f_2\|_1 ,$$

where

$$\|f\|_1 \coloneqq \int_{-\infty}^{+\infty} |f(x)| \mathrm{d}x$$

is the L_1 norm of f. In the following definitions, $\mathcal{F}$ is a class of probability distributions, and g is a distribution not necessarily in $\mathcal{F}$. Denote the set $\{1, 2, ..., m\}$ by $[m]$. All logarithms are in the natural base. For a function g and a class of distributions $\mathcal{F}$, we define

$$\mathrm{OPT}(\mathcal{F}, g) \coloneqq \inf_{f \in \mathcal{F}} \|f - g\|_1$$

Definition 5.1 (ε-approximation, (ε, C)-approximation)**.** *A distribution $\hat{g}$ is an* ε-approximation *for g if $\|\hat{g} - g\|_1 \leq \varepsilon$. A distribution $\hat{g}$ is an* (ε, C)-approximation *for g with respect to $\mathcal{F}$ if*

$$\|\hat{g} - g\|_1 \leq C \times \mathrm{OPT}(\mathcal{F}, g) + \varepsilon$$

Definition 5.2 (PAC-Learning Distributions, Realizable Setting)**.** *A distribution learning method is called a* (realizable) PAC-learner *for $\mathcal{F}$ with sample complexity $m_{\mathcal{F}}(\varepsilon, \delta)$, if for all distribution $g \in \mathcal{F}$ and all $\varepsilon, \delta > 0$, given ε, δ, and a sample of size $m_{\mathcal{F}}(\varepsilon, \delta)$, with probability at least $1 - \delta$ outputs an ε-approximation of g.*

Definition 5.3 (PAC-Learning Distributions, Agnostic Setting)**.** *For $C > 0$, a distribution learning method is called a C-*agnostic PAC-learner for *$\mathcal{F}$ with sample complexity $m_{\mathcal{F}}^{C}(\varepsilon, \delta)$, if for all distributions g and all $\varepsilon, \delta > 0$, given ε, δ, and a sample of size $m_{\mathcal{F}}^{C}(\varepsilon, \delta)$, with probability at least $1 - \delta$ outputs an (ε, C)-approximation of g.*[2]

[2] Note that in some papers, only the case $C \leq 1$ is called agnostic learning, and the case $C > 1$ is called semi-agnostic learning.

Clearly, a C-agnostic PAC-learner (for any constant C) is also a realizable PAC-learner, with the same error parameter ε. Conversely a realizable PAC-learner can be thought of an ∞-agnostic PAC-learner.

5.1.1 Learning Mixture Models

Let Δ_n denote the n-dimensional simplex:

$$\Delta_n \ = \{(w_1, \ldots, w_n) : w_i \geq 0, \sum_{i=1}^{k} w_i = 1\}$$

Definition 5.4. *Let $\mathcal{F}$ be a class of probability distributions. Then the class of k-*mixtures *of $\mathcal{F}$, written $\mathcal{F}^k$, is defined as*

$$\mathcal{F}^k := \left\{ \sum_{i=1}^{k} w_i f_i : (w_1, \ldots, w_k) \in \Delta_k, f_1, \ldots, f_k \in \mathcal{F} \right\}.$$

5.2 Related Work

In the computer science literature, PAC learning of distributions was introduced by [54]; we refer the reader to [42] for a recent survey. A closely related line of research in statistics (in which more emphasis is on sample complexity) is density estimation, for which the book by [39] is an excellent resource.

One approach for studying the sample complexity of learning a class of distributions is bounding the VC-dimension of its associated Yatracos class (see Definition 6.3), and applying results such as Theorem 6.10. (These VC-dimensions have mainly been studied for the purpose of proving generalization bounds for neural networks with sigmoid activation functions.) In particular, the VC-dimension bound of [7, Theorem 8.14] – which is based on the work of [53] – implies a sample complexity upper bound of $O((k^4d^2 + k^3d^3)/\varepsilon^2)$ for PAC learning mixtures of axis-aligned Gaussians, and an upper bound of $O(k^4d^4/\varepsilon^2)$ for PAC learning mixtures of general Gaussians (both results hold in the more general agnostic setting).

A sample complexity upper bound of $O(d^2k^3 \log^2 k/\varepsilon^4)$ for learning mixtures of Gaussians in the realizable setting was proved in [46, Theorem A.1] (the running time of their algorithm is not polynomial). Our algorithm is motivated by theirs, but we have introduced several new ideas in the algorithm and in the analysis, which has resulted in improving the sample complexity bound by a factor of k^2.

For mixtures of spherical Gaussians, a polynomial time algorithm for the realizable setting with sample complexity $O(dk^9 \log^2(d)/\varepsilon^4)$ was proposed in [79, Theorem 11]. We improve their sample complexity by a factor of $\widetilde{O}(k^8)$. In the special case of $d = 1$, a non-proper agnostic polynomial time algorithm with the optimal sample complexity of $\widetilde{O}(k/\varepsilon^2)$ was given in [33], and a proper agnostic algorithm with the same sample complexity and better running time was given in [63].

An important question is finding polynomial time algorithms for learning distributions. To the best of our knowledge, no polynomial time algorithm for learning mixtures of general Gaussians is known. See [46] for the state-of-the-art results. Another important setting is computational complexity in the agnostic learning, see, e.g., [41] for some positive results.

A related line of research is parameter estimation for mixtures of Gaussians, see, e.g., [34, 26, 76], who gave polynomial time algorithms for this problem assuming certain separability conditions (these algorithms are polynomial in the dimension and the error tolerance but exponential in the number of components). Recall that parameter estimation is a more difficult problem and any algorithm for parameter estimation requires some separability assumptions for the target Gaussians, whereas for density estimation no such assumption is needed. E.g., consider the case that $k = 2$ and the two components are identical; then there is no way to learn their mixing weights.

We finally remark that characterizing the sample complexity of learning a class of distributions in general is an open problem, even for the realizable (i.e., non-agnostic) case (see [42, Open Problem 15.1]).

Chapter 6

Learning Mixture Models with/without Advice

In this chapter, we consider a scenario in which we are given a method for learning a class of distributions (e.g., Gaussians). Then, we ask whether we can use it, as a black box, to come up with an algorithm for learning a mixture of such distributions (e.g., mixture of Gaussians). We will show that the answer to this question is affirmative.

We propose a generic method for learning mixture models. Roughly speaking, we show that by going from learning a single distribution from a class to learning a mixture of k distributions from the same class, the sample complexity is multiplied by a factor of at most $(k \log^2 k)/\varepsilon^2$. This result is general, and yet it is surprisingly tight in many important cases. In this dissertation, we assume that the algorithm knows the number of components k.

As a demonstration, we show that our method provides a better sample complexity upper bound for learning mixtures of Gaussians than the state of the art. In particular, for learning mixtures of k Gaussians in $\mathbb{R}^d$, our method requires $\widetilde{O}(d^2k/\varepsilon^4)$ samples, improving by a factor of k^2 over the $\widetilde{O}(d^2k^3/\varepsilon^4)$ bound of [46]. Furthermore, for the special case of mixtures of axis-aligned Gaussians, we provide an upper bound of $\widetilde{O}(dk/\varepsilon^4)$, which is the first optimal bound with respect to k and d up to logarithmic factors, and improves upon the $\widetilde{O}(dk^9/\varepsilon^4)$ bound of [79], which is only shown for the subclass of spherical Gaussians.

We also consider a related setting, where in addition to receiving a sample, the algorithm can ask queries about it. In particular, the algorithm can ask whether two instances were generated from the same component (i.e., same-cluster queries of the previous chapter). We show that using this kind of query, one can devise a computationally efficient method for learning mixtures.

6.1 Contributions

Let $\mathcal{F}$ be a class of probability distributions, and let $\mathcal{F}^k$ denote the class of k-mixtures of elements of $\mathcal{F}$. In our main result, Theorem 6.1, assuming the existence of a method for learning $\mathcal{F}$ with sample complexity $m_{\mathcal{F}}(\varepsilon)$, we provide a method for learning $\mathcal{F}^k$ with sample complexity $O(k \log^2 k \cdot m_{\mathcal{F}}(\varepsilon)/\varepsilon^2)$. Our mixture learning algorithm has the property that, if the $\mathcal{F}$-learner is proper, then the $\mathcal{F}^k$-learner would be proper as well (i.e., the learner will always output a member of $\mathcal{F}^k$). Furthermore, the algorithm works in the more general agnostic setting provided that the base learners are agnostic learners.

We provide several applications of our main result. In Theorem 6.4, we show that the class of mixtures of k axis-aligned Gaussians in $\mathbb{R}^d$ is PAC-learnable with sample complexity $O(kd \log^2 k/\varepsilon^4)$ (see Theorem 6.5). This bound is tight in terms of k and d up to logarithmic factors. In Theorem 6.7, we show that the class of mixtures of k Gaussians in $\mathbb{R}^d$ is PAC-learnable setting with sample complexity $O(kd^2 \log^2 k/\varepsilon^4)$. Finally, in Theorem 6.9, we prove that the class of mixtures of k log-concave distributions over $\mathbb{R}^d$ is PAC-learnable using $\widetilde{O}(d^{(d+5)/2}\varepsilon^{-(d+9)/2}k)$ samples. To the best of our knowledge, this is the first upper bound on the sample complexity of learning this class.

Additionally, we show that if the learner has access to same-cluster queries, then learning can be done in a computationally efficient manner. We show that Gaussian mixture models can be learned in polynomial time using $O(d^2 k \log k/\varepsilon^2)$ samples and $O(d^2 k^2 \log k/\varepsilon^2)$ same-cluster queries. This is an interesting result, as recently there have been some works suggesting that (without queries) there is not much hope for efficient learning of mixtures of Gaussians (see [46] for precise statements). Therefore, in the spirit of the results of the previous chapter, there is a trade-off between the computational and the information-theoretic aspects of learning for the case of mixture learning as well.

6.2 Learning Mixture Models

Assume that we have a method to PAC-learn $\mathcal{F}$. Does this mean that we can PAC-learn $\mathcal{F}^k$? And if so, what is the sample complexity of this task? Our main theorem gives an affirmative answer to the first question, and provides a bound for sample complexity of learning $\mathcal{F}^k$.

Theorem 6.1. *Assume that $\mathcal{F}$ has a C-agnostic PAC-learner with sample complexity $m_{\mathcal{F}}^C(\varepsilon, \delta) = \lambda(\mathcal{F}, \delta)/\varepsilon^\alpha$ for some $C > 0$, $\alpha \geq 1$ and some function $\lambda(\mathcal{F}, \delta) = \Omega(\log(1/\delta))$. Let $\mathcal{F}_\rho^k$ be the set of k-mixtures whose all components are ρ-close to $\mathcal{F}$. Then there exists a PAC-learner for $\mathcal{F}_\rho^k$ that finds a density with error at most $3C\rho + \varepsilon$ and requires $m_{\mathcal{F}_\rho^k}(\varepsilon, \delta) =$*

$$O\left(\frac{\lambda(\mathcal{F},\frac{\delta}{3k})k\log k}{\varepsilon^{\alpha+2}}\right) = O\left(\frac{k\log k\cdot m_{\mathcal{F}}(\varepsilon,\frac{\delta}{3k})}{\varepsilon^2}\right)$$

samples.

We immediately obtain the following corollary.

Corollary 6.1. *Assume that $\mathcal{F}$ has a realizable PAC-learner with sample complexity $m_{\mathcal{F}}(\varepsilon,\delta) = \lambda(\mathcal{F},\delta)/\varepsilon^\alpha$ for some $\alpha \geq 1$ and some function $\lambda(\mathcal{F},\delta) = \Omega(\log(1/\delta))$. Then there exists a realizable PAC-learner for the class $\mathcal{F}^k$ requiring $m_{\mathcal{F}^k}(\varepsilon,\delta) =$*

$$O\left(\frac{\lambda(\mathcal{F},\frac{\delta}{3k})k\log k}{\varepsilon^{\alpha+2}}\right) = O\left(\frac{k\log k\cdot m_{\mathcal{F}}(\varepsilon,\frac{\delta}{3k})}{\varepsilon^2}\right)$$

samples.

Some remarks:

1. Our mixture learning algorithm has the property that, if the $\mathcal{F}$-learner is proper, then the $\mathcal{F}^k$-learner is proper as well.

2. The computational complexity of the resulting algorithm is exponential in the number of required samples.

3. The condition $\lambda(\mathcal{F},\delta) = \Omega(\log(1/\delta))$ is a technical condition that holds for all interesting classes $\mathcal{F}$.

4. One may wonder about tightness of this theorem. In Theorem 2 in [79], it is shown that if $\mathcal{F}$ is the class of spherical Gaussians, we have $m_{\mathcal{F}^k}^{O(1)}(\varepsilon,\delta) = \Omega(km_{\mathcal{F}}(\varepsilon,\delta/k))$, therefore, the factor of k is necessary in general. However, it is not clear whether the additional factor of $\log k/\varepsilon^2$ in the theorem is tight.

5. The constant 3 (in the $3C\rho$-agnostic result) comes from [39, Theorem 6.3] (see Theorem 6.2), and it is not clear whether it is necessary. If we allow for randomized algorithms (which produce a random distribution whose expected distance to the target is bounded by ε), then the constant can be improved to 2, see [68, Theorem 22].

6. It may be possible to extend this result for learning mixture models in the agnostic setting, where each component is not necessarily ρ-close to the base class (but the target distribution is still ρ-close to the class of mixtures). However, our original proof turned out to be incorrect, therefore, we weakened the statement of the theorem. I would like to thank Yaoliang Yu for finding the flaw.

In the rest of this section we prove Theorem 6.1. Let g be the true data generating distribution, and let

$$g^* = \arg\min_{f \in \mathcal{F}^k} \|g - f\|_1 \text{ and } \rho = \|g^* - g\|_1 = \mathrm{OPT}(\mathcal{F}^k, g)\ . \tag{6.1}$$

We have

$$g = \sum_{i \in [k]} w_i G_i,$$

where each G_i is a probability distribution. Let $\rho_i \coloneqq \mathrm{OPT}(\mathcal{F}, G_i)$, and by the assumption we have

$$\sum_{i \in [k]} w_i \rho_i \le \rho. \tag{6.2}$$

The idea now is to learn each of the G_i's separately using the agnostic learner for $\mathcal{F}$. We will view g as a mixture of k distributions $G_1, G_2, \ldots, G_k$.

For proving Theorem 6.1, we will use the following theorem on learning finite classes of distributions, which immediately follows from [39, Theorem 6.3] and a standard Chernoff bound.

Theorem 6.2. *Suppose we are given M candidate distributions $f_1, \ldots, f_M$ and we have access to i.i.d. samples from an unknown distribution g. Then there exists an algorithm that given the f_i's and $\varepsilon > 0$, takes $\log(3M^2/\delta)/2\varepsilon^2$ samples from g, and with probability $\ge 1 - \delta/3$ outputs an index $j \in [M]$ such that*

$$\|f_j - g\|_1 \le 3 \min_{i \in [M]} \|f_i - g\|_1 + 4\varepsilon\ .$$

We now describe an algorithm that with probability $\ge 1 - \delta$ outputs a distribution with L_1 distance $13\varepsilon + 3C\rho$ to g (the error parameter is 13ε instead of ε just for convenience of the proof; it is clear that this does not change the order of magnitude of sample complexity). The algorithm, whose pseudocode is shown in Figure 6.1, has two main steps. In the first step we generate a set of candidate distributions, such that at least one of them is $(3\varepsilon + \rho)$-close to g in L_1 distance. These candidates are of the form $\sum_{i=1}^{k} \widehat{w}_i \widehat{G}_i$, where the $\widehat{G}_i$'s are extracted from samples and are estimates for the real components G_i, and the $\widehat{w}_i$'s come from a fixed discretization of Δ_k, and

```
Input: $k, \varepsilon, \delta$ and an i.i.d. sample $S$
0. Let $\widehat{W}$ be an $(\varepsilon/k)$-cover for $\Delta_k$ in $\ell_\infty$ distance.
1. $\mathcal{C} = \emptyset$ (set of candidate distributions)
2. For each $(\widehat{w}_1, \ldots, \widehat{w}_k) \in \widehat{W}$ do:
   3. For each possible partition of $S$ into
         $A_1, A_2, ..., A_k$:
      4. Provide $A_i$ to the $\mathcal{F}$-learner, and let $\widehat{G}_i$
            be its output.
      5. Add the candidate distribution
            $\sum_{i\in[k]} \widehat{w}_i \widehat{G}_i$ to $\mathcal{C}$.
6. Apply the algorithm for finite classes (Theorem 6.2) to $\mathcal{C}$ and output its result.
```

Figure 6.1: Algorithm for learning the mixture class $\mathcal{F}^k$

are estimates for the real mixing weights w_i. In the second step, we use Theorem 6.2 to obtain a distribution that is $(13\varepsilon + 3C\rho)$-close to g.

We start with describing the first step. We take

$$s = \max\left\{\frac{2k\lambda(\mathcal{F}, \delta/3k)}{\varepsilon^\alpha}, \frac{16k\log(3k/\delta)}{\varepsilon}\right\} \tag{6.3}$$

i.i.d. samples from g. Let S denote the set of generated points. Note that $\lambda(\mathcal{F}, \delta) = \Omega(\log(1/\delta))$ implies

$$s = O(k\lambda(\mathcal{F}, \delta/3k) \times \varepsilon^{-\alpha}).$$

Let $\widehat{W}$ be an ε/k-cover for Δ_k in ℓ_∞ distance of cardinality $(k/\varepsilon + 1)^k$. That is, for any $x \in \Delta_k$ there exists $w \in \widehat{W}$ such that $\|w - x\|_\infty \leq \varepsilon/k$. This can be obtained from a grid in $[0,1]^k$ of side length ε/k, which is an ε/k-cover for $[0,1]^k$, and projecting each of its points onto Δ_k.

By an *assignment*, we mean a function $A : S \to [k]$. The role of an assignment is to "guess" each sample point is coming from which component, by mapping them to a component index. For each pair $(A, (\widehat{w}_1, \ldots, \widehat{w}_k))$, where A is an assignment and $(\widehat{w}_1, \ldots, \widehat{w}_k) \in \widehat{W}$, we generate a candidate distribution as follows: let $A^{-1}(i) \subseteq S$ be those sample points that are assigned to component i. For each $i \in [k]$, we provide the set $A^{-1}(i)$ of samples to our $\mathcal{F}$-learner, and the learner provides us with a distribution $\widehat{G}_i$. We add the distribution $\sum_{i\in[k]} \widehat{w}_i \widehat{G}_i$ to the set of candidate distributions.

Lemma 6.1. *With probability $\geq 1 - 2\delta/3$, at least one of the generated candidate distributions is $(3\varepsilon + C\rho)$-close to g.*

Before proving the lemma, we show that it implies our main result, Theorem 6.1. By the lemma, we obtain a set of candidates such that at least one of them is $(3\varepsilon + C\rho)$-close to g (with failure probability $\leq 2\delta/3$). This step takes $s = O(k\lambda(\mathcal{F}, \delta/3k) \times \varepsilon^{-\alpha})$ many samples. Then, we apply Theorem 6.2 to output one of those candidates that is $(13\varepsilon + 3C\rho)$-close to g (with failure probability $\leq \delta/3$), therefore using $\log(3M^2/\delta)/2\varepsilon^2$ additional samples. Note that the number of generated candidate distributions is $M = k^s \times (1 + k/\varepsilon)^k$. Hence, in the second step of our algorithm, we take

$$\begin{aligned}\log(3M^2/\delta)/2\varepsilon^2 &= O\left(\frac{\lambda(\mathcal{F}, \delta/3k)k\log k}{\varepsilon^{\alpha+2}}\right)\\ &= O\left(\frac{m_{\mathcal{F}}(\varepsilon, \delta/3k)k\log k}{\varepsilon^2}\right)\end{aligned}$$

additional samples. The proof is completed noting the total failure probability is at most δ by the union bound.

We now prove Lemma 6.1. We will use the following concentration inequality, which holds for any binomial random variable X (see [75, Theorem 4.5(2)]):

$$\Pr\{X < \mathbf{E}X/2\} \leq \exp(-\mathbf{E}X/8)\,. \tag{6.4}$$

Say a component i is *negligible* if

$$w_i \leq \frac{8\log(3k/\delta)}{s}$$

Let $L \subseteq [k]$ denote the set of negligible components. Let i be a non-negligible component. Note that, the number of points coming from component i is binomial with parameters s and w_i and thus has mean sw_i, so (6.4) implies that, with probability at least $1 - \delta/3k$, S contains at least $w_i s/2$ points from i. Since we have k components in total, the union bound implies that, with probability at least $1 - \delta/3$, uniformly for all $i \notin L$, S contains at least $w_i s/2$ points from component i.

Now consider the pair $(A, (\widehat{w}_1, \ldots, \widehat{w}_k))$ such that A assigns samples to their correct indices, and has the property that $|\widehat{w}_i - w_i| \leq \varepsilon/k$ for all $i \in [k]$. We claim that the resulting candidate distribution is $(3\varepsilon + C\rho)$-close to g.

Let $\widehat{G}_1, \ldots, \widehat{G}_k$ be the distributions provided by the learner. For each $i \in [k]$ define

$$\varepsilon_i := \left(\frac{2\lambda(\mathcal{F}, \delta/3k)}{w_i s}\right)^{1/\alpha}$$

For any $i \notin L$, since there exists at least $w_i s/2$ samples for component i, and since

$$w_i s/2 = \lambda(\mathcal{F}, \delta/3k)\varepsilon_i^{-\alpha} = m_{\mathcal{F}}(\varepsilon_i, \delta/3k) \,,$$

we are guaranteed that $\|\widehat{G}_i - G_i\|_1 \leq C\rho_i + \varepsilon_i$ with probability $1 - \delta/3k$ (recall that each G_i is ρ_i-close to the class $\mathcal{F}$). Therefore, $\|\widehat{G}_i - G_i\|_1 \leq C\rho_i + \varepsilon_i$ holds uniformly over all $i \notin L$, with probability $\geq 1 - \delta/3$. Note that since $\alpha \geq 1$, the function $w_i^{1-1/\alpha}$ is concave in w_i, so by Jensen's inequality we have

$$\sum_{i\in[k]} w_i^{1-1/\alpha} \leq k \left((\sum_{i\in[k]} w_i/k)^{1-1/\alpha} \right) = k^{1/\alpha} \,,$$

hence

$$\begin{aligned}\sum_{i\notin L} w_i\varepsilon_i &= \left(\frac{2\lambda(\mathcal{F}, \delta/3k)}{s}\right)^{1/\alpha} \sum_{i\notin L} w_i^{1-1/\alpha} \\ &\leq \left(\frac{2k\lambda(\mathcal{F}, \delta/3k)}{s}\right)^{1/\alpha} .\end{aligned}$$

Also recall from (6.2) that $\sum_{i\in[k]} w_i\rho_i \leq \rho$. Proving the lemma is now a matter of careful applications of the triangle inequality:

$$\begin{aligned}&\left\|\sum_{i\in[k]} \widehat{w}_i\widehat{G}_i - g\right\|_1 = \left\|\sum_{i\in[k]} \widehat{w}_i\widehat{G}_i - \sum_{i\in[k]} w_iG_i\right\|_1 \\ &\leq \left\|\sum_{i\in[k]} w_i(\widehat{G}_i - G_i)\right\|_1 + \left\|\sum_{i\in[k]} (\widehat{w}_i - w_i)\widehat{G}_i\right\|_1 \\ &\leq \left\|\sum_{i\in L} w_i(\widehat{G}_i - G_i)\right\|_1 + \left\|\sum_{i\notin L} w_i(\widehat{G}_i - G_i)\right\|_1 \\ &\qquad + \sum_{i\in[k]} |\widehat{w}_i - w_i| \left\|\widehat{G}_i\right\|_1 \\ &\leq 2\sum_{i\in L} w_i + \sum_{i\notin L} w_i(\varepsilon_i + C\rho_i) + \sum_{i\in[k]} \varepsilon/k \times 1 \\ &\leq 2k \times \frac{8\log(3k/\delta)}{s} + \left(\frac{2k\lambda(\mathcal{F}, \delta/3k)}{s}\right)^{1/\alpha} + C\rho + \varepsilon \\ &\leq \varepsilon + \varepsilon + \varepsilon + C\rho \,,\end{aligned}$$

where for the last inequality we used the definition of s in (6.3). This completes the proof of Lemma 6.1.

6.3 Learning Mixtures of Gaussians

Gaussian Mixture Models (GMMs) are probably the most widely studied mixture classes with numerous applications; yet, the sample complexity of learning this class is not fully understood, especially when the number of dimensions is large. In this section, we will show that our method for learning mixtures can improve the state of the art for learning GMMs in terms of sample complexity. In the following, $\mathcal{N}_d(\mu, \Sigma)$ denotes a Gaussian density function defined over $\mathbb{R}^d$, with mean μ and covariance matrix Σ.

6.3.1 Mixtures of Axis-Aligned Gaussians

A Gaussian is called *axis-aligned* if its covariance matrix Σ is diagonal. The class of axis-aligned Gaussian Mixtures is an important special case of GMMs that is thoroughly studied in the literature (e.g. [48]).

Theorem 6.3. *Let $\mathcal{F}$ denote the class of d-dimensional axis-aligned Gaussians. Then $\mathcal{F}$ is 3-agnostic PAC-learnable with $m^3_{\mathcal{F}}(\varepsilon, \delta) = O((d + \log(1/\delta))/\varepsilon^2)$.*

We defer the proof of this result to Section 6.7. Combining this theorem with Theorem 6.1 we obtain the following result:

Theorem 6.4. *The class $\mathcal{F}^k$ of mixtures of k axis-aligned Gaussians in $\mathbb{R}^d$ is PAC-learnable with sample complexity $m_{\mathcal{F}^k}(\varepsilon, \delta) = O(kd \log k \log(k/\delta)/\varepsilon^4)$.*

This theorem improves the upper bound of $O(dk^9 \log^2(d/\delta)/\varepsilon^4)$ proved in [79, Theorem 11] for spherical Gaussians in the realizable setting. Spherical Gaussians are special cases of axis-aligned Gaussians in which all eigenvalues of the covariance matrix are equal, i.e., Σ is a multiple of the identity matrix. The following minimax lower bound (i.e., worst-case on all instances) on the sample complexity of learning mixtures of spherical Gaussians is proved in the same paper.

Theorem 6.5 (Theorem 2 in [79])**.** *The class $\mathcal{F}^k$ of mixtures of k axis-aligned Gaussians in $\mathbb{R}^d$ in the realizable setting has $m_{\mathcal{F}^k}(\varepsilon, 1/2) = \Omega(dk/\varepsilon^2)$.*

Therefore, our upper bound of Theorem 6.4 is optimal in terms of dependence on d and k (up to logarithmic factors) for axis-aligned Gaussians.

6.3.2 Mixtures of General Gaussians

For general Gaussians, we have the following result.

Theorem 6.6. *Let $\mathcal{F}$ denote the class of d-dimensional Gaussians. Then, $\mathcal{F}$ is 3-agnostic PAC-learnable with $m^3_{\mathcal{F}}(\varepsilon, \delta) = O((d^2 + \log(1/\delta))/\varepsilon^2)$.*

We defer the proof of this result to Section 6.7. Combining this theorem with Theorem 6.1, we obtain the following result:

Theorem 6.7. *The class $\mathcal{F}^k$ of mixtures of k Gaussians in $\mathbb{R}^d$ is PAC-learnable with sample complexity $m_{\mathcal{F}^k}(\varepsilon, \delta) = O(kd^2 \log k \log(k/\delta)/\varepsilon^4)$.*

This improves by a factor of k^2 the upper bound of $O(k^3 d^2 \log k/\varepsilon^4)$ in the realizable setting, proved in [46, Theorem A.1].

Note that Theorem 6.5 gives a lower bound of $\Omega(kd/\varepsilon^2)$ for $m_{\mathcal{F}^k}(\varepsilon, \delta)$, hence the dependence of Theorem 6.7 on k is optimal (up to logarithmic factors). However, there is a factor of d/ε^2 between the upper and lower bounds.

6.4 Learning Mixtures of Gaussians with Queries

The learning algorithm that we used to prove Theorem 6.7 about learning mixtures of Gaussians is not efficient. In fact, its time complexity is exponential in k, d and $1/\epsilon$. Nevertheless, the existence of a polynomial-time algorithm (with respect to k, d, and $1/\epsilon$) is a major open problem. Therefore, it would be interesting to see if the use of same-cluster queries can help here, as it did in Chapter 4.

Assume that the distribution learning method has access to a same-cluster oracle, and can therefore ask whether two instances were generated from the same component or not. We showed in Section 4.7 that k same-cluster queries can be used to answer to a cluster-assignment query. In other words, we would like to know if having access to a cluster-assignment oracle—one that can tell us the index of the component which generated an instance—can provide us with a polynomial-time algorithm for learning GMMs.

Note that in the proof of Theorem 6.1, we created an exponential number of candidate distributions and then chose between them. In fact, this was the reason that our algorithm was computationally inefficient. However, this would not be necessary if the method has access to the "labels" (i.e., the component indices) of the given instances. In particular, the algorithm does not need to guess the labels (as they are given) or the mixing weights (as they can be substituted by their empirical values). Therefore, together with the fact that the base learners are efficient (i.e., a single Gaussian distribution can be learned in polynomial-time in the realizable setting, see [10, Appendix B]), we would have a polynomial time for learning GMMs in the realizable case. This result can even be extended to the non-realizable setting, using the fact that there are efficient and robust methods for learning a single Gaussian distribution (see, [43, 60, 44]).

6.5 Learning Mixtures of Log-Concave Distributions

A probability density function over $\mathbb{R}^d$ is log-concave if its logarithm is a concave function. The following result about the sample complexity of learning log-concave distributions is the direct consequence of the recent work of [45].

Theorem 6.8. *Let $\mathcal{F}$ be the class of distributions corresponding to the set of all log-concave densities over $\mathbb{R}^d$. Then $\mathcal{F}$ is 3-agnostic PAC learnable using $m^3(\varepsilon, \delta) = O((d/\varepsilon)^{(d+5)/2} \log^2(1/\varepsilon))$ samples.*

Using Theorem 6.1, we come up with the first result about the sample complexity of learning mixtures of log-concave distributions.

Theorem 6.9. *The class of mixtures of k log-concave distributions over $\mathbb{R}^d$ is PAC-learnable using $\widetilde{O}(d^{(d+5)/2}\varepsilon^{-(d+9)/2}k)$ samples.*

6.6 Conclusions

We studied PAC learning of classes of distributions that are in the form of mixture models, and proposed a generic approach for learning such classes in the cases where we have access to a black box method for learning a single-component distribution. We showed that by going from one component to a mixture model with k components, the sample complexity is multiplied by a factor of at most $(k \log^2 k)/\varepsilon^2$.

Furthermore, as a corollary of this general result, we provided upper bounds for the sample complexity of learning GMMs and axis-aligned GMMs—$O(kd^2 \log^2 k/\varepsilon^4)$ and $O(kd \log^2 k/\varepsilon^4)$ respectively. Both of these results improve upon the state of the art in terms of dependence on k and d.

It is worthwhile to note that for the case of GMMs, the dependence of our bound is $1/\varepsilon^4$. Therefore, proving an upper bound of kd^2/ε^2 remains open.

Also, note that our result can be readily applied to the general case of mixtures of the exponential family. Let $\mathcal{F}_d$ denote the d-parameter exponential family. Then the VC-dimension of the corresponding Yatracos class (see Definition 6.3) is $O(d)$ (see Theorem 8.1 in [39]) and therefore by Theorem 6.10, the sample complexity of PAC learning $\mathcal{F}_d$ is $O(d/\varepsilon^2)$. Finally, applying Theorem 6.1 gives a sample complexity upper bound of $\widetilde{O}(kd/\varepsilon^4)$ for learning $\mathcal{F}_d^k$.

6.7 Appendix: Proofs of Theorems 6.3 and 6.6

We follow the general methodology of [39] to prove upper bounds on the sample complexity of learning Gaussian distributions. The idea is to first connect distribution learning to the VC-dimension of a class of a related set system (called the Yatracos class of the corresponding distribution family), and then provide upper bounds on VC-dimension of this system. Our Theorem 6.10 gives an upper bound for the sample complexity of agnostic learning, given an upper bound for the VC-dimension of the Yatracos class. We remark that a variant of this result, without explicit dependence on the failure probability, is proved implicitly in [33] and also appears explicitly in [45, Lemma 6].

Definition 6.1 ($\mathcal{A}$-Distance)**.** *Let $\mathcal{A} \subset 2^X$ be a class of subsets of domain X. Let p and q be two probability distributions over X. Then the* $\mathcal{A}$-distance *between p and q is defined as*

$$\|p - q\|_{\mathcal{A}} \coloneqq \sup_{A \in \mathcal{A}} |p(A) - q(A)|$$

Definition 6.2 (Empirical Distribution)**.** *Let $S = \{x_i\}_{i=1}^m$ be a sequence of members of X. The* empirical distribution *corresponding to this sequence is defined by $\hat{p}_S(x) = \sum_{i=1}^m \frac{\mathbb{1}\{x=x_i\}}{m}$.*

The following lemma is a well known refinement of the uniform convergence theorem, see, e.g., [7, Theorem 4.9].

Lemma 6.2. *Let p be a probability distribution over X. Let $\mathcal{A} \subseteq 2^X$ and let v be the VC-dimension of $\mathcal{A}$. Then, there exist universal positive constants c_1, c_2, c_3 such that*

$$\mathbf{Pr}_{S \sim p^m}\{\|p - \hat{p}_S\|_{\mathcal{A}} \geq \varepsilon\} \leq \exp(c_1 + c_2 v - c_3 m \varepsilon^2)\,.$$

Definition 6.3 (Yatracos class)**.** *For a class $\mathcal{F}$ of functions from X to $\mathbb{R}$, their* Yatracos class *is the family of subsets of X defined as*

$$\mathcal{Y}(\mathcal{F}) \coloneqq \{\{x \in X : f_1(x) \geq f_2(x)\} \text{ for some } f_1, f_2 \in \mathcal{F}\}$$

Observe that if $f, g \in \mathcal{F}$ then $\|f - g\|_{TV} = \|f - g\|_{\mathcal{Y}(F)}$.

Definition 6.4 (Empirical Yatracos Minimizer)**.** *Let $\mathcal{F}$ be a class of distributions over domain X. The* empirical Yatracos minimizer *is defined as $L^{\mathcal{F}} : \cup_{m=1}^{\infty} X^m \to \mathcal{F}$ satisfying*

$$L^{\mathcal{F}}(S) = \arg\min_{q \in \mathcal{F}} \|q - \hat{p}_S\|_{\mathcal{Y}(\mathcal{F})}.$$

Theorem 6.10 (PAC Learning Families of Distributions)**.** *Let $\mathcal{F}$ be a class of probability distributions, and let $S \sim p^m$ be an i.i.d. sample of size m generated from an arbitrary probability distribution p, which is not necessarily in $\mathcal{F}$. Then with probability at least $1 - \delta$ we have*

$$\|p - L^{\mathcal{F}}(S)\|_{TV} \leq 3\,\mathrm{OPT}(\mathcal{F}, p) + \alpha\sqrt{\frac{v + \log\frac{1}{\delta}}{m}}$$

where v is VC-dimension of $\mathcal{Y}(\mathcal{F})$, and $\mathrm{OPT}(\mathcal{F}, p) = \inf_{q^* \in \mathcal{F}} \|q^* - p\|_{TV}$, and α is a universal constant. In particular, in the realizable setting $p \in \mathcal{F}$, we have

$$\|p - L^{\mathcal{F}}(S)\|_{TV} \leq \alpha\sqrt{\frac{v + \log\frac{1}{\delta}}{m}}$$

Remark 6.1. *The L_1 distance is precisely twice the total variation distance.*

Proof. Let $q^* = \arg\min_{q \in \mathcal{F}} \|p - q\|_{TV}$, so $\|q^* - p\|_{\mathcal{Y}(\mathcal{F})} \leq \|q^* - p\|_{TV} = \mathrm{OPT}(\mathcal{F}, p)$. Since $L^{\mathcal{F}}(S), q^* \in \mathcal{F}$ we have $\|L^{\mathcal{F}}(S) - q^*\|_{TV} = \|L^{\mathcal{F}}(S) - q^*\|_{\mathcal{Y}(\mathcal{F})}$. By Lemma 6.2, with probability $\geq 1 - \delta$ we have $\|p - \hat{p}_S\|_{\mathcal{A}} \leq \alpha\sqrt{(v + \log\frac{1}{\delta})/m}$ for some universal constant α. Also, since $L^{\mathcal{F}}(S)$ is the empirical minimizer of the $\mathcal{Y}(\mathcal{F})$-distance, we have $\|L^{\mathcal{F}}(S) - \hat{p}_S\|_{\mathcal{Y}(\mathcal{F})} \leq \|q^* - \hat{p}_S\|_{\mathcal{Y}(\mathcal{F})}$. The proof follows from these facts combined with multiple applications of the triangle inequality:

$$\begin{aligned}
\|p - L^{\mathcal{F}}(S)\|_{TV} &\leq \|L^{\mathcal{F}}(S) - q^*\|_{TV} + \|q^* - p\|_{TV} \\
&= \|L^{\mathcal{F}}(S) - q^*\|_{\mathcal{Y}(\mathcal{F})} + \mathrm{OPT}(\mathcal{F}, p) \\
&\leq \|L^{\mathcal{F}}(S) - \hat{p}_S\|_{\mathcal{Y}(\mathcal{F})} + \|\hat{p}_S - q^*\|_{\mathcal{Y}(\mathcal{F})} + \mathrm{OPT}(\mathcal{F}, p) \\
&\leq \|q^* - \hat{p}_S\|_{\mathcal{Y}(\mathcal{F})} + \big(\|\hat{p}_S - p\|_{\mathcal{A}} + \|p - q^*\|_{\mathcal{Y}(\mathcal{F})}\big) + \\
&\mathrm{OPT}(\mathcal{F}, p) \leq \big(\|q^* - p\|_{\mathcal{Y}(\mathcal{F})} + \|p - \hat{p}_S\|_{\mathcal{A}}\big) + \\
&\|p - \hat{p}_S\|_{\mathcal{Y}(\mathcal{F})} + 2\,\mathrm{OPT}(\mathcal{F}, p) \\
&\leq \|q^* - p\|_{TV} + 2\|p - \hat{p}_S\|_{\mathcal{Y}(\mathcal{F})} + 2\,\mathrm{OPT}(\mathcal{F}, p) \\
&\leq 2\alpha\sqrt{\frac{v + \log\frac{1}{\delta}}{m}} + 3\,\mathrm{OPT}(\mathcal{F}, p)\,.
\end{aligned}$$

□

Theorem 6.10 provides a tool for proving upper bounds on the sample complexity of distribution learning. To prove Theorems 6.6 and 6.3, it remains to show upper bounds on the VC dimensions of the Yatracos class of (axis-aligned) Gaussian densities.

For classes $\mathcal{F}$ and $\mathcal{G}$ of functions, let

$$\mathrm{NN}(\mathcal{G}) \coloneqq \{\{x : f(x) \geq 0\} \text{ for some } f \in \mathcal{G}\}$$

and

$$\Delta\mathcal{F} \coloneqq \{f_1 - f_2 : f_1, f_2 \in \mathcal{F}\}\,,$$

and notice that

$$\mathcal{Y}(\mathcal{F}) = \mathrm{NN}(\Delta\mathcal{F}).$$

We upper bound the VC-dimension of $\mathrm{NN}(\Delta\mathcal{F})$ via the following well known result in statistical learning theory, see, e.g., [39, Lemma 4.2].

Theorem 6.11 (Dudley)**.** *Let $\mathcal{G}$ be an n-dimensional vector space of real-valued functions. Then $VC(\mathrm{NN}(\mathcal{G})) \leq n$.*

Now let h be an indicator function for an arbitrary element in $\mathrm{NN}(f_1 - f_2)$, where f_1, f_2 are densities of (axis-aligned) Gaussians. Then h is a $\{0, 1\}$-valued function and we have:

$$\begin{aligned} h(x) &= \mathbb{1}\{\mathcal{N}(\mu_1, \Sigma_1) > \mathcal{N}(\mu_2, \Sigma_2)\} \\ &= \mathbb{1}\{\alpha_1 \exp(\frac{-1}{2}(x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1)) > \\ &\qquad \alpha_2 \exp(\frac{-1}{2}(x-\mu_2)^T\Sigma_2^{-1}(x-\mu_2))\} \\ &= \mathbb{1}\{(x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1) \\ &\qquad -(x-\mu_2)^T\Sigma_2^{-1}(x-\mu_2) - \log\frac{\alpha_2}{\alpha_1} > 0\}\,. \end{aligned}$$

The inner expression is a quadratic form, and the linear dimension of all quadratic functions is $O(d^2)$. Furthermore, for axis-aligned Gaussians, Σ_1 and Σ_2 are diagonal, and therefore, the inner function lies in an $O(d)$-dimensional space of functions spanned by $\{1, x_1, \ldots, x_d, x_1^2, \ldots, x_d^2\}$. Hence, by Dudley's theorem, we have the required upper bound (d or d^2) on the VC-dimension of the Yatracos classes. Finally, Theorems 6.6 and 6.3 follow from applying Theorem 6.10 to the class of (axis-aligned) Gaussian distributions.

Chapter 7

Learning Mixture Models via Compression

Learning about a probability distribution from a sample generated by that distribution is a fundamental task. In this chapter, we follow the same formulation of Chapters 5 and 6 for distribution learning. Determining the sample complexity of learning with respect to a general class of distributions in this model is an open problem (see [42, Open Problem 15.1]).

In this chapter, we study the class of k-mixtures of axis-aligned Gaussians over $\mathbb{R}^d$ (i.e., distributions whose probability density functions (PDFs) are convex combinations of k axis-aligned Gaussians' PDFs). Distribution learning with respect to this class, as well as the related class of mixtures of spherical Gaussians (those whose covariance matrices are multiples of the identity matrix), has been studied extensively [79, 41, 48]. Surprisingly, the best possible sample complexity of learning with respect to these classes is still unknown.

The state-of-the-art in terms of k and d is the result that we proved in the previous chapter (Theorem 6.1), which provides an upper bound of $\widetilde{O}(kd/\epsilon^4)$ for learning with respect to the class of mixtures of axis-aligned Gaussians. On a high level, the idea was to start with an i.i.d. sample of size $\widetilde{O}(kd/\epsilon^2)$, and then partition this sample in every possible way into k subsets. Then, roughly $O(k^{kd/\epsilon^2})$ "candidate distributions" were generated based on those partitions. The problem was then reduced to learning with respect to a finite class of candidates. However, the exponential dependence of the number of candidates on $1/\epsilon$ made the final bound loose in terms of $1/\epsilon$. It turns out that there is no easy way to remove that exponential dependence.

As the main technical result of this chapter, we prove that the class of k-mixtures of axis-aligned Gaussian distributions over $\mathbb{R}^d$ can be learned using $\widetilde{O}(kd/\epsilon^2)$ samples. This is the first result that, up to logarithmic factors, matches the known lower bound of $\Omega(kd/\epsilon^2)$ [79].

We prove our main result by introducing a new form of *sample compression*. On a high-level, we show that if we are able to "encode" members of a class of distributions using only a few of the samples generated from them, then we can get an upper bound on the sample complexity of learning with respect to that class. In particular, by proposing a compression scheme for the class of mixtures of axis-aligned Gaussians, we come up with a nearly sharp upper bound on the sample complexity of learning with respect to that class.

Let us emphasize again that we address the problem of *density estimation* rather than that of *parameter estimation* (see [42, Section 15.2] for their difference). This is motivated by the fact that in many applications, identifying the parameters is not the goal *per se*; instead, it suffices to have a good approximation of the target distribution.

The approach we adopt for proving the upper bound is algorithmic. However, our focus is not on computational efficiency. In particular, the bound is proved using a sample-efficient method whose running time is exponential in terms of the Euclidean dimension and the number of components of the mixture.

7.1 Contributions

In this chapter, we introduce a novel method for learning distributions via a form of *sample compression*. Given a class of distributions, assume that there is a method for "compressing" the samples that are generated by any distribution in the class. Further, assume that there exists a fixed *decoder* for the class, such that given the compressed set of instances, it approximately recovers the original distribution. In this case, if the size of the compressed set is guaranteed to be small, we show that the sample complexity of learning that class is small as well.

We say that a class *admits* (t, m) compression if there exists a compression scheme such that after generating m samples from any distribution in the class, we are guaranteed, with high probability, to have a subset of size at most t of that sample, from which the decoder reconstructs the original distribution.

We will also formalize a related but stronger notion of *robust compression*, where the target distribution is supposed to be encoded using samples that are not necessarily generated from the target itself, but are generated from a distribution that is close to the target (see Definition 7.2).

We prove that robust compression implies agnostic/robust learning. In particular, we show that if a class admits (t, m) robust compression, then the sample complexity of agnostic learning with respect to this class is roughly $O(m + t \log m/\epsilon^2)$. Note that m and t can be functions of ϵ, the accuracy parameter.

We also prove some closure properties of compression. Namely, we prove that if a base class admits compression, then the class of k-mixtures of that base class, as well as the class of products of the base class, are compressible (Lemmas 7.1 and 7.2). Consequently, it will suffice to provide a compression scheme for one-dimensional Gaussian distributions in order to obtain a compression scheme for mixtures of axis-aligned Gaussians (and therefore, to be able to bound the sample complexity of learning that class).

As the final step, we prove that the class of one-dimensional Gaussian distributions admits $(O(1), O(1/\epsilon))$ robust compression. Constructing this constant-size robust compression scheme ultimately enables us to prove a sharp bound for the sample complexity of learning, in terms of the dependence on ϵ.

The above results together imply an upper bound of $\widetilde{O}(kd/\epsilon^2)$ for learning k-mixtures of axis-aligned Gaussian distributions over $\mathbb{R}^d$ (and consequently, for the subset of mixtures of spherical Gaussian distributions). This is the first upper bound for this class that is tight in d, k, and ϵ, and matches, up to logarithmic factors, the minimax lower bound of $\Omega(kd/\epsilon^2)$ [79].

The *compression* framework that we introduce is generic, and can be used to prove sample complexity upper bounds for other classes of distributions as well.

7.2 Distribution Compression Schemes

For a distribution g, $S \sim g^m$ means that S is an i.i.d. sample of size m generated from g. Let $\mathcal{F}$ be a class of distributions over a domain Z.

Definition 7.1 (distribution decoder)**.** *A* distribution decoder *for $\mathcal{F}$ is a deterministic function $\mathcal{J} : \bigcup_{n=0}^{\infty} Z^n \times \bigcup_{n=0}^{\infty} \{0,1\}^n \to \mathcal{F}$, which takes a finite sequence of elements of Z and a finite sequence of bits, and outputs a member of $\mathcal{F}$.*

Definition 7.2. *[robust distribution compression schemes] Let $t_1, t_2, m : (0,1) \to \mathbb{Z}_{\geq 0}$ be functions, and let $r \geq 0$. We say that $\mathcal{F}$ admits (t_1, t_2, m) r-robust compression if there exists a decoder $\mathcal{J}$ for $\mathcal{F}$ such that for any distribution $g \in \mathcal{F}$, and for any distribution q on Z with $\|g - q\|_1 \leq r$, the following holds:*

> *For any $\varepsilon \in (0,1)$, if $S \sim q^{m(\varepsilon)}$, then with probability at least $2/3$, there exists a sequence L of at most $t_1(\varepsilon)$ elements of S, and a sequence B of at most $t_2(\varepsilon)$ bits, such that $\|\mathcal{J}(L, B) - g\|_1 \leq \varepsilon$.*

Essentially, the definition asserts that with high probability, there should be a (small) subset of S and some (small number of) additional bits, from which g can be reconstructed. We say that the distribution g is "encoded" with L and B, and in general we would like to have a compression scheme of a small size. This compression scheme is called “robust” since one wants to reconstruct g based on a sample that is generated from q rather than g itself. We will mainly consider constant values of r, and therefore q can be quite dissimilar to g.

Remark 7.1. *In the next sections, we will see that $(t_1 + t_2)$, the total number of bits and instances used for compression, is the core quantity in the analysis. Therefore, we sometimes use the notation of (t, m) compression rather than the triplet notation, which means that the total number of bits and instances together is bounded by t. An “efficient” encoding will be one in which the size of the compression scheme, $t(\varepsilon)$ is either bounded by a constant, or at most logarithmically dependent on $1/\varepsilon$.*

Remark 7.2. *In the definition above we required the probability of existence of L and B to be at least 2/3, but note that if this holds, one can boost this probability to $1 - \delta$ by generating a sample of size $m(\varepsilon) \log(1/\delta)$.*

7.3 Robust Compression Implies Agnostic Learning

In this section, we show that if a class of distributions can be compressed, then it can be learned; thus we build the connection between robust compression and agnostic learning. We will need the following useful result about PAC-learning of finite classes of distributions, which immediately follows from [39, Theorem 6.3] and a standard Chernoff bound. Essentially, it suggests that finite classes of size M can be 3-learned in the agnostic setting using $O(\log(M/\delta)/\epsilon^2)$ samples. Denote by $[M]$ the set $\{1, 2, ..., M\}$.

Theorem 7.1. *Suppose we are given M candidate distributions $f_1, \ldots, f_M$ and we have access to i.i.d. samples from an unknown distribution g. Then there exists an algorithm that given the f_i's and $\varepsilon > 0$, takes $\log(3M^2/\delta)/2\varepsilon^2$ samples from g, and with probability $\geq 1 - \delta/3$ outputs an index $j \in [M]$ such that*

$$\|f_j - g\|_1 \leq 3 \min_{i \in [M]} \|f_i - g\|_1 + 4\varepsilon \ .$$

Theorem 7.2. *Suppose $\mathcal{F}$ admits (t_1, t_2, m) r-robust compression. Let $t(\varepsilon) := d(\varepsilon/6) + t(\varepsilon/6)$. Then $\mathcal{F}$ can be $\max\{3, 2/r\}$-learned in the agnostic setting using*

$$O\left(m(\frac{\varepsilon}{6}) \log \frac{1}{\delta} + \frac{t(\varepsilon) \log(m(\frac{\varepsilon}{6}) \log(1/\delta)) + \log(1/\delta)}{\varepsilon^2}\right)$$
$$= \widetilde{O}\left(m(\frac{\varepsilon}{6}) + \frac{t(\varepsilon)}{\varepsilon^2}\right) \textit{ samples.}$$

Proof. Let q be the target distribution that the samples are being generated from. Let $\alpha = \inf_{f \in \mathcal{F}} \|f - q\|_1$ be the approximation error of q with respect to $\mathcal{F}$. The goal of the learner is to find a distribution $\hat{h}$ such that $\|\hat{h} - q\|_1 \leq \max\{3, 2/r\} \cdot \alpha + \varepsilon$.

First, consider the case $\alpha \leq r$. In this case, we develop a learner that finds a distribution $\hat{h}$ such that $\|\hat{h} - q\|_1 \leq 3\alpha + \varepsilon$. Let $g \in \mathcal{F}$ be a distribution such that $\|g - q\|_1 \leq \alpha + \frac{\varepsilon}{12}$ (such a g exists by the definition of α). By assumption, $\mathcal{F}$ admits (t_1, t_2, m) compression. Let $\mathcal{J}$ denote the corresponding decoder. Given ε, the learner first asks for an i.i.d. sample $S \sim q^{m(\varepsilon/6)\cdot\log(2/\delta)}$. By the definition of robust compression, we know that with probability at least $1 - \delta/2$, there exist $L \in S^{t_1(\varepsilon/6)}$ and $B \in \{0,1\}^{t_2(\varepsilon/6)}$ such that $\|\mathcal{J}(L, B) - g\| \leq \varepsilon/6$ (see Remark 7.2).

The learner is of course unaware of L and B. However, given the sample S, it can try all of the possibilities for L and B and create a candidate set of distributions. More concretely, let $H = \{\mathcal{J}(L, B) : L \in S^{t_1(\varepsilon/6)}, B \in \{0,1\}^{t_2(\varepsilon/6)}\}$. Note that

$$\begin{aligned} |H| &\leq (m(\varepsilon/6)\log(2/\delta))^{t_1(\varepsilon/6)} 2^{t_2(\varepsilon/6)} \\ &\leq (m(\varepsilon/6)\log(2/\delta))^{t(\varepsilon)}. \end{aligned}$$

Since H is finite, we can use the algorithm of Theorem 7.1 to find a good candidate $\hat{h}$ from H. In particular, we set the accuracy parameter in Theorem 7.1 to be $\varepsilon/16$ and the confidence parameter to be $\delta/2$. In this case, Theorem 7.1 requires

$$\frac{\log(6|H|^2/\delta)}{2(\varepsilon/16)^2} = O\left(\frac{t(\varepsilon)\log(m(\frac{\varepsilon}{6})\log(\frac{1}{\delta})) + \log(\frac{1}{\delta})}{\varepsilon^2}\right)$$

additional samples—which is $\widetilde{O}(t(\varepsilon)/\varepsilon^2)$—, and its output $\hat{h}$ will be an (ε,3)-approximation of q:

$$\begin{aligned} \|\hat{h} - q\|_1 &\leq 3\|h^* - q\|_1 + 4\frac{\varepsilon}{16} \\ &\leq 3(\|h^* - g\|_1 + \|g - q\|_1) + \frac{\varepsilon}{4} \\ &\leq 3(\varepsilon/6 + (\alpha + \varepsilon/12)) + \frac{\varepsilon}{4} \leq 3\alpha + \varepsilon. \end{aligned}$$

Note that the above procedure uses $\widetilde{O}(m(\varepsilon/6) + \frac{t(\varepsilon)}{\varepsilon^2})$ samples, and the probability of failure is at most δ (i.e., the probability of either H not containing a good h^*, or the failure of Theorem 7.1 in choosing a good candidate among H, is bounded by $\delta/2 + \delta/2 = \delta$).

The other case, $\alpha > r$, is trivial: the learner outputs some distribution $\widehat{h}$. Since $\widehat{h}$ and q are density functions, we have $\|\widehat{h} - q\|_1 \leq 2 < \frac{2}{r} \cdot \alpha < \max\{3, 2/r\} \cdot \alpha + \varepsilon$. □

7.4 Robust Compression of Products of Distributions

In this section, we show that if a class $\mathcal{F}$ of distributions can be compressed, then the class of distributions that are formed by taking products of distributions in $\mathcal{F}$ can also be compressed. Recall that if $p_1, \ldots, p_d$ are d distributions over domains $Z_1, \ldots, Z_d$, then their product $\prod_{i=1}^d p_i$ is a distribution over $\prod_{i=1}^d Z_i$ defined as $(\prod_{i=1}^d p_i)(\prod_{i=1}^d A_i) = \prod_{i=1}^d p_i(A_i)$ for any measurable $A_1 \subseteq Z_1, \ldots, A_d \subseteq Z_d$. For a class $\mathcal{F}$ of distributions, we define $\mathcal{F}^d \coloneqq \left\{ \prod_{i=1}^d p_i : p_1, \ldots, p_d \in \mathcal{F} \right\}$. The following proposition is standard; see the appendix of this chapter for a proof.

Proposition 7.1. *For $i \in [d]$, let p_i and q_i be probability distributions over the same domain Z. Then $\|\Pi_{i=1}^d p_i - \Pi_{i=1}^d q_i\|_1 \leq \sum_{i=1}^d \|p_i - q_i\|_1$.*

Lemma 7.1 (Compressing Product Distributions)**.** *If $\mathcal{F}$ admits $(t_1(\varepsilon), t_2(\varepsilon), m(\varepsilon))$ r-robust compression, then $\mathcal{F}^d$ admits $(dt_1(\varepsilon/d), d.t_2(\varepsilon/d), m(\varepsilon/d)\log 3d)$ r-robust compression.*

Proof. Let $G = \Pi_{i=1}^d g_i$ be an arbitrary element of $\mathcal{F}^d$. Let Q be an arbitrary distribution over Z^d, subject to $\|G - Q\|_1 \leq r$. Let $q_1, \ldots, q_d$ be the marginal distributions of Q on the d components. First, we claim that $\|q_j - g_j\|_1 \leq r$ for each $j \in [d]$. For, suppose there exists some $j \in [d]$ with $\|q_j - g_j\|_1 > r$. By symmetry, we may assume $j = 1$. This means $\|q_1 - g_1\|_{TV} > r/2$, so there exists $A \subset Z$ such that $p_1(A) - q_1(A) > r/2$. This means

$$\begin{aligned} \|Q - G\|_1/2 &= \|Q - G\|_{TV} \\ &\geq \mathbf{Pr}_Q(A \times Z \times \cdots \times Z) - \mathbf{Pr}_G(A \times Z \times \cdots \times Z) \\ &= q_1(A) - g_1(A) > r/2, \end{aligned}$$

which contradicts $\|G - Q\|_1 \leq r$. Hence, we have $\|q_j - g_j\|_1 \leq r$ for all $j \in [d]$.

We know that $\mathcal{F}$ admits (t_1, t_2, m) r-robust compression. Call the corresponding decoder $\mathcal{J}$, and let $m_0 = m(\varepsilon/d)\log(3d)$, and $S \sim Q^{m_0}$. The goal is then to encode an ε-approximation of G using $d.t_1(\varepsilon/d)$ elements of S and $d.t_2(\varepsilon/d)$ bits.

Note that each element of S is an n-dimensional vector. For each $i \in [d]$, let $S_i \in Z^{m_0}$ be the set of the i-th components of elements of S. By definition of q_i, we have $S_i \sim q_i^{m_0}$ for each i. Thus, for each $i \in [d]$, since $\|q_i - g_i\| \leq r$, with probability at least $1 - 1/3d$ there exists a sequence L_i of at most $t_1(\varepsilon/d)$ elements of S_i, and a sequence B_i of at most $t_2(\varepsilon/d)$ bits, such that $\|\mathcal{J}(L_i, B_i) - g_i\|_1 \leq \varepsilon/d$. By the union bound, this assertion holds for all $i \in [d]$, with probability at least $2/3$. We may encode these $L_1, \ldots, L_d, B_1, \ldots, B_d$ using $d.t_1(\varepsilon/d)$ elements of S and $d.t_2(\varepsilon/d)$ bits. Our decoder for $\mathcal{F}^d$ then extracts $L_1, \ldots, L_d, B_1, \ldots, B_d$ from these elements and bits, and then outputs $\prod_{i=1}^d \mathcal{J}(L_i, B_i) \in \mathcal{F}^d$. Finally, Proposition 7.1 gives $\|\Pi_{i=1}^d \mathcal{J}(L_i, B_i) - G\|_1 \leq \sum_{i=1}^d \|\mathcal{J}(L_i, B_i) - g_i\|_1 \leq d \times \varepsilon/d \leq \varepsilon$, completing the proof. □

7.5 Compression of Mixtures of Distributions

In this section, we show that if a class $\mathcal{F}$ of distributions can be compressed, then the class of distributions that are formed by taking mixtures of distributions in $\mathcal{F}$ can also be compressed. We start by defining mixtures. Let Δ_n denote the n-dimensional simplex, $\Delta_n \coloneqq \{(w_1, \ldots, w_n) : w_i \geq 0, \sum_{i=1}^n w_i = 1\}$.

Definition 7.3. *Let $\mathcal{F}$ be a class of probability distributions. Then the class of k-*mixtures *of $\mathcal{F}$, written k-mix($\mathcal{F}$), is defined as*

$$k\text{-mix}(\mathcal{F}) \coloneqq \left\{ \sum_{i=1}^{k} w_i f_i : (w_1, \ldots, w_k) \in \Delta_k, f_j \in \mathcal{F} \right\}.$$

Lemma 7.2 (Compressing Mixtures)**.** *Let $m(\varepsilon)$ be an invertible function and suppose that $xm^{-1}(km(\varepsilon/3)x)$ is a concave function of x. If $\mathcal{F}$ admits (t_1, t_2, m) compression for t_1 and t_2 that are independent of ε, then k-*mix*$(\mathcal{F})$ admits $(kt_1, kt_2 + k\log_2(4k/\varepsilon), s(\varepsilon))$ compression, where*

$$s(\varepsilon) = \max\left\{2\log(6k)km(\varepsilon/3), 48k\log(6k)/\varepsilon\right\}.$$

Remark 7.3. *Any function of the form $m(\varepsilon) = C\varepsilon^{-\alpha}$ with $\alpha \geq 1$ satisfies the first assumption of the lemma.*

Proof. Suppose $g^* \in k$-mix$(\mathcal{F})$ is the distribution to be compressed. Thus we have $g^* = \sum_{i\in[k]} w_i f_i$ with each $f_i \in \mathcal{F}$ and $(w_1, \ldots, w_k) \in \Delta_k$.

The samples from g^* can be partitioned into k parts, so that samples from the i-th part have distribution f_i. We compress each of the parts individually.

Moreover, we compress the mixing weights $w_1, \ldots, w_k$ using bits, as follows. Consider an $(\varepsilon/3k)$-cover in ℓ_∞ for Δ_k, of size $(1 + 3k/\varepsilon)^k$. Such a cover can be obtained from a mesh of grid-size $\varepsilon/3k$, and projecting each of its point onto Δ_k. Let $(\widehat{w}_1, \ldots, \widehat{w}_k)$ be an element in the cover that has

$$\|(\widehat{w}_1, \ldots, \widehat{w}_k) - (w_1, \ldots, w_k)\|_\infty \leq \varepsilon/3k,$$

then, $w_i - \widehat{w}_i \leq \varepsilon/3k$ for all i. Moreover, the particular element $(\widehat{w}_1, \ldots, \widehat{w}_k)$ of the cover can be encoded using $\log_2((1 + 3k/\varepsilon)^k) \leq k\log_2(4k/\varepsilon)$ bits.

For any $i \in [k]$, we say component i is *negligible* if

$$w_i \leq 8\log(6k)/s.$$

By a standard Chernoff bound together with a union bound over the k components, with probability at least $5/6$, for each non-negligible component i, we have at least $w_i s/2$ samples from i. Let

$$\beta = km(\varepsilon/3), \quad \varepsilon_i = m^{-1}(w_i\beta) = m^{-1}(w_i km(\varepsilon/3)).$$

Let i be a non-negligible component. Since $s \geq 2\log(6k)\beta$ we have $w_i s/2 \geq \log(6k) \times m(\varepsilon_i)$, so since $\mathcal{F}$ admits (t_1, t_2, m) compression and $f_i \in \mathcal{F}$, with probability at least $1 - 1/6k$ there exists t_1 samples from part i and t_2 bits, from which the decoder can construct a distribution $\widehat{f_i}$ with $\|f_i - \widehat{f_i}\|_1 \leq \varepsilon_i$ (recall that we have assumed that t_1 and t_2 are constant and thus independent of ε_i). Using a union bound over the k components, this is true uniformly over all non-negligible components, with probability at least $5/6$. (Note that, for negligible components i, there is no guarantee about $\widehat{f_i}$.) Hence, given the mixing weights $\widehat{w}_1, \ldots, \widehat{w}_k$, the decoder outputs $\sum \widehat{w}_i \widehat{f_i}$.

Thus to complete the proof of the lemma, we need only show that $\| \sum w_i f_i - \sum \widehat{w}_i \widehat{f_i}\|_1 \leq \varepsilon$, which we prove by showing two inequalities.

First, let $L \subseteq [k]$ denote the set of negligible components. Since $s \geq 48k\log(6k)/\varepsilon$, if $i \in L$ then $w_i \leq 8\log(6k)/s \leq \varepsilon/6k$, and thus

$$\sum_{i \in L} w_i \leq k \times \varepsilon/6k \leq \varepsilon/6.$$

Second, since the function $h(x) = xm^{-1}(\beta x)$ is concave in x, by Jensen's inequality we have $\frac{1}{k}\sum h(w_i) \leq h(\sum w_i/k) = h(1/k)$, which gives

$$\begin{aligned}\sum_{i \in [k]} w_i \varepsilon_i &= \sum_{i \in [k]} w_i m^{-1}(\beta w_i) \leq k \times (1/k) m^{-1}(\beta(1/k)) \\ &= m^{-1}(\beta/k) = \varepsilon/3,\end{aligned}$$

where for the last equality we used the definition of $\beta = km(\varepsilon/3)$. Putting everything together, we obtain

$$\begin{aligned}\left\| \sum_{i \in [k]} (\widehat{w}_i \widehat{f_i} - w_i f_i) \right\|_1 &\leq \left\| \sum_{i \in [k]} w_i (\widehat{f_i} - f_i) \right\|_1 + \\ \left\| \sum_{i \in [k]} (\widehat{w}_i - w_i) \widehat{f_i} \right\|_1 &\leq \left\| \sum_{i \in L} w_i (\widehat{f_i} - f_i) \right\|_1 + \\ \left\| \sum_{i \notin L} w_i (\widehat{f_i} - f_i) \right\|_1 &+ \sum_{i \in [k]} |\widehat{w}_i - w_i| \left\| \widehat{f_i} \right\|_1 \\ &\leq 2 \sum_{i \in L} w_i + \sum_{i \notin L} w_i \varepsilon_i + \sum_{i \in [k]} \varepsilon/3k \times 1 \\ &\leq \varepsilon/3 + \varepsilon/3 + \varepsilon/3 = \varepsilon,\end{aligned}$$

completing the proof of the lemma. □

7.6 Robust Compression of Univariate Gaussian Distributions

In this section, we show that the class of 1-dimensional Gaussians can be compressed. This is the core result of our analysis, and will ultimately enable us to show that the class of mixtures of axis-aligned Gaussians can be compressed, hence can be learned as well.

Let $\mathcal{N}(\mu, \sigma)$ denote a 1-dimensional Gaussian distribution with mean μ and standard deviation σ. We will need the following lemma, bounding the L_1 distance of two Gaussians in terms of their parameters. The proof can be found in the appendix.

Lemma 7.3. *There exist a constant c_2 such that for any $\mu, \sigma, \widehat{\mu}, \widehat{\sigma}$ with $|\widehat{\mu} - \mu| \leq \varepsilon_1 \sigma$ and $|\widehat{\sigma} - \sigma| \leq \varepsilon_2 \sigma$ and $\varepsilon_1, \varepsilon_2 \in (0, 1/2)$ we have $\|\mathcal{N}(\mu, \sigma) - \mathcal{N}(\widehat{\mu}, \widehat{\sigma})\|_1 \leq c_2(\varepsilon_1 + \varepsilon_2)$.*

Any vector $(p_1, \dots, p_n) \in \Delta_n$ induces a discrete probability distribution over $[n]$ defined by $\mathbf{Pr}(i) := p_i$.

Lemma 7.4. *Let $(p_1, \dots, p_{2n+1}) \in \Delta_{2n+1}$ and $(q_1, \dots, q_{2n+1}) \in \Delta_{2n+1}$ be discrete probability distributions with ℓ_1 distance between them $\leq t$. Suppose we have $2n+1$ bins, numbered 1 to $2n+1$. We throw m balls in these bins, where each ball chooses a bin independently according to q_i. We pair bin 1 with bin 2, bin 3 with bin 4, …, and bin $2n-1$ with bin $2n$; so bin $2n+1$ is unpaired. The probability that, for all pairs of bins, at most one them gets a ball, is not more than*

$$2^n \left(t/2 + p_{2n+1} + \sum_{i=1}^{n} \max\{p_{2i-1}, p_{2i}\} \right)^m$$

Proof. Let $P_1 = \{1, 2\}$, $P_2 = \{3, 4\}$, ..., $P_n = \{2n-1, 2n\}$, and let $\mathcal{A} := \{A \subset [2n] : |A \cap P_i| = 1 \; \forall i \in [n]\}$. Clearly $|\mathcal{A}| = 2^n$. For any $A \in \mathcal{A}$, let E_A be the event that, the first ball does not choose a bin in A, and let F_A be the event that, none of the balls chooses a bin in A. Then,

$$\begin{aligned} \mathbf{Pr}[E_A] &= \sum_{i \in [2n+1] \setminus A} q_i \\ &\leq \|p - q\|_{TV} + \sum_{i \in [2n+1] \setminus A} p_i \leq t/2 + \sum_{i \notin A} p_i \\ &\leq t/2 + p_{2n+1} + \sum_{i=1}^{n} \max\{p_{2i-1}, p_{2i}\}, \end{aligned}$$

and so $\mathbf{Pr}[F_A] = \mathbf{Pr}[E_A]^m \leq (t/2 + p_{2n+1} + \sum_{i=1}^{n} (p_{2i-1} \vee p_{2i}))^m$. Finally, observe that, if for each pair of bins, at most one them gets a ball, then there exists at least one $A \in \mathcal{A}$, such that none of the balls chooses a bin in A. The lemma is thus proved by applying the union bound over all events $\{F_A\}_{A \in \mathcal{A}}$. □

Theorem 7.3. *The class of all Gaussian distributions over the real line admits* $(4, 1, O(1/\varepsilon))$ 0.773*-robust compression.*

Proof. Let q be any distribution (not necessarily a Gaussian) such that there exists a Gaussian $g = \mathcal{N}(\mu, \sigma)$ with $\|q - g\|_1 \leq r \leq 0.773$. Our goal is to encode g using samples generated from q. Let $m = C/\varepsilon$ for a large enough constant C to be determined, and let $S \sim q^m$ be an i.i.d. sample. The idea is to approximately encode μ and σ using only four elements of S and a single bit.

We start by defining the decoder $\mathcal{J}$. Our proposed decoder takes as input four instances $x_1, x_2, y_1, y_2 \in \mathbb{R}$, and one bit $b \in \{0, 1\}$. The decoder then outputs a Gaussian pdf based on the following rule:

$$\mathcal{J}(x_1, x_2, y_1, y_2, b) = \begin{cases} \mathcal{N}(\frac{x_1+x_2}{2}, \frac{|y_1-y_2|}{3}) & : \text{ if } b = 1 \\ \\ \mathcal{N}(\frac{x_1+x_2}{2}, |y_1 - y_2|) & : \text{ if } b = 0 \end{cases}$$

Our goal is thus to show that, with probability at least $2/3$, there exists $x_1, x_2, y_1, y_2 \in S$ and $b \in \{0, 1\}$ so that $\|\mathcal{J}(x_1, x_2, y_1, y_2, b) - g\| \leq \varepsilon$.

Let $M = 1/\varepsilon$ and partition the interval $[-2\sigma, 2\sigma)$ into $4M$ subintervals of length $\varepsilon\sigma$. Enumerate these intervals as I_1 to I_{4M}, i.e., $I_i = [-2\sigma + (i-1)(\varepsilon\sigma), -2\sigma + i(\varepsilon\sigma))$. Also let $I_{4M+1} = \mathbb{R} \setminus \bigcup_{i=1}^{4M} I_i$. We state two claims which will imply the theorem, and which will be proved later.

Claim 1. With probability at least $5/6$, there exist $y_1, y_2 \in S$ such that at least one of the following two conditions holds: (a) $y_1 \in I_i$ and $y_2 \in I_{i+M}$ for some $i \in \{M+1, 2M+2, ..., 2M\}$. In this case, we let $b = 0$, and so $\mathcal{J}(x_1, x_2, y_1, y_2, b)$ will have standard deviation $|y_1 - y_2|$.
(b) $y_1 \in I_i$ and $y_2 \in I_{i+3M}$ for some $i \in [M]$. In this case, we let $b = 1$, and so $\mathcal{J}(x_1, x_2, y_1, y_2, b)$ will have standard deviation $\frac{|y_1-y_2|}{3}$.
Also, if both cases of (a) and (b) happen, we will go with the first rule. Note that if Claim 1 holds, and $\hat{\sigma}$ is the standard deviation of $\mathcal{J}(x_1, x_2, y_1, y_2, b)$, then we will have $|\hat{\sigma} - \sigma| \leq \varepsilon\sigma$.

Claim 2. With probability at least $5/6$, there exist $x_1, x_2 \in S$ such that $x_1 \in I_i$ and $x_2 \in I_{4M-i+1}$ for some $i \in [2M]$. If so, $\mathcal{J}(x_1, x_2, y_1, y_2, b)$ will have mean $\frac{x_1+x_2}{2} =: \hat{\mu}$.

Also note that if Claim 2 holds, then $|\hat{\mu} - \mu| \leq \varepsilon\sigma$. Therefore, if both claims hold, Lemma 7.3 gives $\mathcal{J}(x_1, x_2, y_1, y_2, b) = \mathcal{N}(\hat{\mu}, \hat{\sigma})$ would be a $c_2\varepsilon$-approximation for $\mathcal{N}(\mu, \sigma) = g$, for some constant c_2. In other words, g can be approximately reconstructed, up to error $c_2\varepsilon$, using only four data points (i.e., $\{x_1, x_2, y_1, y_2\}$) from a sample S of size $O(1/\varepsilon)$ and a single bit b (the definition of robust compression requires an ε-compression. For getting this, one just needs to refine the partition by a constant factor, which multiplies M by a constant factor, and as we will see below, this will only multiply m by a constant factor). Note also that the probability of existence of such four points is at least $1 - (1 - 5/6) - (1 - 5/6) \geq 2/3$.

Therefore, it remains to prove Claim 1 and Claim 2. We prove Claim 1, and the proof for Claim 2 is similar.

View the sets $I_1, \ldots, I_{4M}, I_{4M+1}$ as bins, and consider the i.i.d. samples as balls landing in these bins according to q. Let $p_i \coloneqq \int_{I_i} g(x)\mathrm{d}x$ and $q_i \coloneqq \int_{I_i} q(x)\mathrm{d}x$ for $i \in [4M+1]$. Note that, by triangle's inequality, the ℓ_1 distance between $(p_1, \ldots, p_{4M+1})$ and $(q_1, \ldots, q_{4M+1})$ is not more than the L_1 distance between g and q, which is at most r. Let $x \vee y \coloneqq \max\{x, y\}$.

We pair the bins as follows: I_i is paired with I_{i+M} for $i \in \{M+1, \ldots, 2M\}$, and I_i is paired with I_{i+3M} for $i \in [M]$. Therefore, by Lemma 7.4, the probability that Claim 1 does not hold can be bounded by

$$\begin{aligned}
&2^{2M}\left(\sum_{i=M+1}^{2M}(p_i \vee p_{i+M}) + \sum_{i=1}^{M}(p_i \vee p_{i+3M}) + p_{4M+1} + \frac{r}{2}\right)^m \\
&= 2^{2M}\left(\sum_{i=\frac{3}{2}M+1}^{\frac{5}{2}M} p_i + \sum_{i=\frac{M}{2}+1}^{M} p_i + \sum_{3M+1}^{\frac{7}{2}M} p_i + p_{4M+1} + \frac{r}{2}\right)^m,
\end{aligned}$$

where in the last step we used the fact that p_i are coming from a Gaussian, and thus $p_1 \leq \cdots \leq p_{2M} = p_{2M+1} \geq \cdots \geq p_{4M}$ (we have also assumed, for simplicity, that M is even). Let $\Phi(A) \coloneqq \mathbf{Pr}_{x\sim\mathcal{N}(0,1)}[x \in A]$. Then we get

$$\begin{aligned}
&\sum_{i=1.5M+1}^{2.5M} p_i + \sum_{i=M/2+1}^{M} p_i + \sum_{3M+1}^{3.5M} p_i + p_{4M+1} + r/2 \\
&= \mathbf{Pr}[\mathcal{N}(\mu, \sigma) \in [\mu - \sigma/2, \mu + \sigma/2]] \\
&+ 2\mathbf{Pr}[N(\mu, \sigma) \in [\mu - 3\sigma/2, \mu - \sigma]] \\
&+ \mathbf{Pr}[N(\mu, \sigma) \notin [\mu - 2\sigma, \mu + 2\sigma]] + r/2 \\
&= \Phi([-0.5, 0.5]) + 2\Phi([-1.5, -1]) + 2\Phi((-\infty, -2]) + \frac{r}{2} \\
&< 0.383 + 0.184 + 0.046 + \frac{r}{2} = 0.613 + r/2 \leq 0.9995.
\end{aligned}$$

Therefore since $M = \Theta(1/\varepsilon)$, by making $m = C/\varepsilon$ for a large enough C, we can make this probability arbitrarily small, completing the proof of Claim 1.

Via a similar argument, the probability that Claim 2 does not hold can be bounded by

$$\begin{aligned}
&2^{2M}\left(\sum_{i=1}^{2M}\max\{p_i, p_{4M-i+1}\} + p_{4M+1} + r/2\right)^m \\
&= 2^{2M}\left(\sum_{i=1}^{2M} p_i + p_{4M+1} + r/2\right)^m \\
&= 2^{2M}\left(\Phi([-1,1]) + \Phi([2,\infty) + r/2\right)^m \\
&< 2^{2M}\left(0.5 + 0.023 + r/2\right)^m < 2^{2M}\left(0.91\right)^m < 1/6,
\end{aligned}$$

for $m = C/\varepsilon$ with a large enough C. □

Remark 7.4. *By using more bits and adding more scales, one can show that 1-dimensional Gaussians admit $(4, b(r), O(1/\varepsilon))$ r-robust compression for any fixed $r < 1$ (the number of required bits and the implicit constant in the O will depend on the value of r), but this will not result in an improvement in the main result of this chapter, Corollary 7.1.*

7.7 Compression of Mixtures of Axis-aligned Gaussians

Theorem 7.4. *The class of mixtures of k axis-aligned Gaussian distributions over $\mathbb{R}^d$ admits $(O(kd), O(kd + k\log(k/\varepsilon)), O((kd\log k\log d)/\varepsilon))$ compression.*

Proof. Let $\mathcal{G}$ denote the set of all 1-dimensional Gaussian distributions. By Theorem 7.3, $\mathcal{G}$ admits $(O(1), O(1), O(1/\varepsilon))$ compression.

By Lemma 7.1, the class $\mathcal{G}^d$ admits $(O(d), O(d), O((d\log d)/\varepsilon))$ compression.

Then, by Lemma 7.2, the class k-mix($\mathcal{G}^d$) admits $(O(kd), O(kd+k\log(k/\varepsilon)), O((kd\log k\log d/\varepsilon)))$ compression. □

Applying Theorem 7.2 we obtain the main result of this chapter.

Corollary 7.1. *The class of mixtures of k axis-aligned Gaussians in $\mathbb{R}^n$ can be learned using $\widetilde{O}(kd/\varepsilon^2)$ many samples.*

We note that this bound is tight up to logarithmic factors, as a minimax (worst-case) lower bound of $\Omega(kd/\varepsilon^2)$ was proved in [79, Theorem 2].

7.8 Compression of Mixtures of General Gaussians

We want mention that the compression framework is powerful enough to be used for determining the sample complexity of mixtures of general Gaussians as well. In fact, we have been able to settle the sample complexity of learning this class up to logarithmic factors [10]. In particular, we have showed that the minimax rate of learning mixtures of k Gaussians in $\mathbb{R}^d$ is $\widetilde{\Theta}(kd^2/\epsilon^2)$. This very recent result is excluded from this thesis.

7.9 Further Discussions

In the context of binary classification, the fully combinatorial notion of Littlestone-Warmuth compression has been shown to be sufficient [64] and necessary [77] for learning. For distribution learning, while we have shown that compression is sufficient, its necessity remains an open problem.

Another related concept to compression is the notion of *core-sets*. In a sense, core-sets can be viewed as a special case of compression, where the decoder is required to be the empirical error minimizer. See the work of [66] for the use core-sets in maximum likelihood estimation. Nevertheless, in our case, the additional flexibility of compression schemes proves to be useful in allowing for a constant-size scheme, and ultimately having a sharper bound in terms of ϵ.

7.10 Appendix: Proofs of Auxiliary Results

Proposition 7.2. *For $i \in [d]$, let p_i and q_i be probability distributions over the same domain Z. Then $\|\Pi_{i=1}^d p_i - \Pi_{i=1}^d q_i\|_1 \leq \sum_{i=1}^d \|p_i - q_i\|_1$.*

Proof. For $i \in [d]$, let p_i and q_i be arbitrary probability distributions over the same domain. We will prove that

$$\|\Pi_{i=1}^n p_i - \Pi_{i=1}^d q_i\|_{TV} \leq \sum_{i=1}^d \|p_i - q_i\|_{TV},$$

and this gives the proposition, since the L_1 distance is precisely twice the total variation distance. By the coupling characterization of the total variation distance, there exist couplings (X_1, Y_1), (X_2, Y_2), ..., (X_d, Y_d), such that for each i we have $X_i \sim p_i$, $Y_i \sim q_i$, and $\mathbf{Pr}[X_i \neq Y_i] =$

$\|p_i - q_i\|_{TV}$. Observe that $(X_1, \ldots, X_d) \sim \prod p_i$ and $(Y_1, \ldots, Y_d) \sim \prod q_i$, hence by the union bound,

$$\|\Pi_{i=1}^d p_i - \Pi_{i=1}^d q_i\|_{TV} \leq \mathbf{Pr}[(X_1, \ldots, X_d) \neq (Y_1, \ldots, Y_d)] \leq \sum \mathbf{Pr}(X_i \neq Y_i) = \sum \|p_i - q_i\|_{TV}.$$

□

Proposition 7.3. *Suppose that g and g^* are distributions with $\|g - g^*\|_1 = \rho$ and $g^* = \sum_{i \in [k]} w_i f_i$, with $(w_1, \ldots, w_k) \in \Delta_k$ and where each f_i is a distribution. Then, we may write $g = \sum_{i \in [k]} w_i G_i$, such that each G_i is a distribution, and for each i we have $\|f_i - G_i\| \leq \rho$.*

Proof. Suppose that g and g^* are distributions with $\|g - g^*\|_1 = \rho$ and $g^* = \sum_{i \in [k]} w_i f_i$, with $(w_1, \ldots, w_k) \in \Delta_k$ and where each f_i is a distribution. Then we want to show that we may write $g = \sum_{i \in [k]} w_i G_i$, such that each G_i is a distribution, and for each i we have $\|f_i - G_i\| \leq \rho$.

Write

$$g = g^* + h = \sum_{i=1}^k w_i f_i + h = \sum_{i=1}^k w_i (f_i + h) \tag{7.1}$$

with $\|h\|_1 = \rho$. Note that $f_i + h$ is not necessarily a density function. Let $\mathcal{D}$ denote the set of density functions, that is, the set of nonnegative functions with unit L_1 norm. Note that this is a convex set. Since projection is a linear operator, by projecting both sides of (7.1) onto $\mathcal{D}$ we find

$$g = \sum_{i=1}^k w_i G_i,$$

where G_i is the L_1 projection of $f_i + h$ onto $\mathcal{D}$ (since $g \in \mathcal{D}$, the projection of g onto $\mathcal{D}$ is itself). Also, since $f_i \in \mathcal{F} \cap \mathcal{D}$ and projection onto a convex set does not increases distances, we have

$$\|f_i - G_i\|_1 \leq \|f_i - (f_i + h)\|_1 = \|h\|_1 = \rho,$$

as required. □

The following lemma is [57, Lemma 4.9].

Lemma 7.5 ([57]). *There is a constant c_1 such that for any $\sigma_1, \sigma_2 \in \mathbb{R}^+$ we have*

$$\|\mathcal{N}(0, \sigma_1) - \mathcal{N}(0, \sigma_2)\|_1 \leq c_1 \left(\frac{\max(\sigma_1, \sigma_2)^2}{\min(\sigma_1, \sigma_2)^2} - 1 \right)$$

Lemma 7.6. *For any $\mu_1, \mu_2 \in \mathbb{R}$ we have*

$$\|\mathcal{N}(\mu_1, 1) - \mathcal{N}(\mu_2, 1)\|_1 \leq |\mu_1 - \mu_2|$$

Proof. Let $D_{KL}(f\|g)$ denote the KL-divergence between f and g. Using Pinsker's inequality, we have

$$\|\mathcal{N}(\mu_1, 1) - \mathcal{N}(\mu_2, 1)\|_1 \leq \sqrt{2D_{KL}(\mathcal{N}(\mu_1, 1)\|\mathcal{N}(\mu_2, 1))} = \sqrt{2(1/2)(\mu_1 - \mu_2)^2} = |\mu_1 - \mu_2|.$$

□

Lemma 7.7. *There exist a constant c_2 such that for any $\mu, \sigma, \widehat{\mu}, \widehat{\sigma}$ with $|\widehat{\mu} - \mu| \leq \varepsilon_1\sigma$ and $|\widehat{\sigma} - \sigma| \leq \varepsilon_2\sigma$ and $\varepsilon_1, \varepsilon_2 \in (0, 1/2)$ we have*

$$\|\mathcal{N}(\mu, \sigma) - \mathcal{N}(\widehat{\mu}, \widehat{\sigma})\|_1 \leq c_2(\varepsilon_1 + \varepsilon_2).$$

Proof. Setting $c_2 = \max\{8c_1, 1\}$, the proof follows from the use of triangle inequality, Lemmas 7.5 and 7.6, and the fact that variation distance is scale invariant (recall that $\varepsilon_1, \varepsilon_2 \in (0, 1/2)$):

$$\begin{aligned}\|\mathcal{N}(\mu, \sigma) - \mathcal{N}(\widehat{\mu}, \widehat{\sigma})\|_1 &\leq \|\mathcal{N}(\mu, \sigma) - \mathcal{N}(\widehat{\mu}, \sigma)\|_1 + \|\mathcal{N}(\widehat{\mu}, \sigma) - \mathcal{N}(\widehat{\mu}, \widehat{\sigma})\|_1 \\ &\leq \varepsilon_1 + 8c_1\varepsilon_2 \leq c_2(\varepsilon_1 + \varepsilon_2).\end{aligned}$$

□

Chapter 8

More Future Directions

We proposed the future directions at the end of the corresponding chapters. In this chapter, we will mention a few *additional* research directions.

Efficient Representation Learning for Semi-Supervised Clustering. As discussed in Chapter 2, in practice, model selection for clustering is often done via *ad hoc* methods. Furthermore, it is impossible to perform model selection without looking at the intended semantics of the domain of interest. Therefore, it is critical to devise user-friendly protocols that enable conveying domain knowledge into the process of model selection for clustering. In Chapter 3 and Chapter 4 we introduced new frameworks for achieving this goal. However, we are still far from the intended gold standard: a semi-supervised clustering method that is provably (i) computationally efficient, (ii) statistically tractable, and (iii) rich enough to capture user knowledge. In particular, the proposed method in Chapter 4 was not rich enough for all applications (in the sense that it assumed the intended clustering has a particular geometric structure). It seems that performing representation learning for clustering is the natural way for addressing the richness issue. Note that the proposed representation learning method in Chapter 3 was not computationally efficient. Therefore, the existence of an efficient representation learning method for semi-supervised clustering (with a guarantee of success given enough advice) remains a major direction of future research.

Semi-supervised Clustering with Noisy Oracles. In Chapter 4, we showed how semi-supervised clustering can be done efficiently using same-cluster queries. To answer to these queries, however, we would need an oracle that could verify whether two instances belong to the same cluster or not. The access to such a perfect oracle however is unrealistic in many applications. Therefore, being able to handle noisy oracles is an important missing feature of our results. Note that this would even allow us to train a classifier to answer to most of the queries (and ask the domain expert only if it could not confidently classify the given instance).

Density Estimation via Compression: More Classes, and More Distances. In Chapter 7 we proposed a method for sample-efficient distribution learning (i.e., density estimation). While we proved our results only for Gaussian distributions and their mixtures, our framework was generic and could be adopted to obtain sample-efficient methods for learning other classes of distributions. Therefore, extending our results to learning other classes of distributions (and distance functions) remains an exciting direction of future research.

References

[1] Jayadev Acharya, Ilias Diakonikolas, Jerry Li, and Ludwig Schmidt. Sample-optimal density estimation in nearly-linear time. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '17, pages 1278–1289, Philadelphia, PA, USA, 2017. Society for Industrial and Applied Mathematics.

[2] Margareta Ackerman, Shai Ben-David, and David Loker. Towards property-based classification of clustering paradigms. In *Advances in Neural Information Processing Systems*, pages 10–18, 2010.

[3] Nir Ailon, Anup Bhattacharya, and Ragesh Jaiswal. Approximate correlation clustering using same-cluster queries. *arXiv preprint arXiv:1712.06865*, 2017.

[4] Nir Ailon, Anup Bhattacharya, Ragesh Jaiswal, and Amit Kumar. Approximate clustering with same-cluster queries. *arXiv preprint arXiv:1704.01862*, 2017.

[5] Babak Alipanahi, Michael Biggs, Ali Ghodsi, et al. Distance metric learning vs. fisher discriminant analysis. In *Proceedings of the 23rd national conference on Artificial intelligence*, pages 598–603, 2008.

[6] Dana Angluin. Queries and concept learning. *Machine learning*, 2(4):319–342, 1988.

[7] Martin Anthony and Peter Bartlett. *Neural network learning: theoretical foundations*. Cambridge University Press, 1999.

[8] Martin Anthony and Peter L Bartlett. *Neural network learning: Theoretical foundations*. cambridge university press, 2009.

[9] Hassan Ashtiani and Shai Ben-David. Representation learning for clustering: A statistical framework. In *Uncertainty in AI (UAI)*, 2015.

[10] Hassan Ashtiani, Shai Ben-David, Christopher Liaw, Abbas Mehrabian, and Yaniv Plan. Settling the sample complexity for learning mixtures of gaussians. *arXiv preprint arXiv:1710.05209*, 2018.

[11] Hassan Ashtiani, Shai Ben-David, and Abbas Mehrabian. Agnostic distribution learning via compression. *arXiv preprint arXiv:1710.05209v1*, 2017.

[12] Hassan Ashtiani, Shai Ben-David, and Abbas Mehrabian. Sample-efficient learning of mixtures. In *AAAI*, 2018.

[13] Hassan Ashtiani and Ali Ghodsi. A dimension-independent generalization bound for kernel supervised principal component analysis. In *Proceedings of The 1st International Workshop on Feature Extraction: Modern Questions and Challenges, NIPS*, pages 19–29, 2015.

[14] Hassan Ashtiani, Shrinu Kushagra, and Shai Ben-David. Clustering with same-cluster queries. In *Advances in neural information processing systems*, pages 3216–3224, 2016.

[15] Pranjal Awasthi, Maria Florina Balcan, and Konstantin Voevodski. Local algorithms for interactive clustering. *Journal of Machine Learning Research*, 18(3):1–35, 2017.

[16] Pranjal Awasthi, Avrim Blum, and Or Sheffet. Center-based clustering under perturbation stability. *Information Processing Letters*, 112(1):49–54, 2012.

[17] Pranjal Awasthi and Reza B Zadeh. Supervised clustering. In *Advances in Neural Information Processing Systems*, pages 91–99, 2010.

[18] Maria-Florina Balcan and Avrim Blum. Clustering with interactive feedback. In *Algorithmic Learning Theory*, pages 316–328. Springer, 2008.

[19] Maria-Florina Balcan, Avrim Blum, and Anupam Gupta. Approximate clustering without the approximation. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1068–1077. Society for Industrial and Applied Mathematics, 2009.

[20] Maria-Florina Balcan, Avrim Blum, and Santosh Vempala. A discriminative framework for clustering via similarity functions. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 671–680. ACM, 2008.

[21] Maria Florina Balcan and Yingyu Liang. Clustering under perturbation resilience. In *Automata, Languages, and Programming*, pages 63–74. Springer, 2012.

[22] Sugato Basu, Arindam Banerjee, and Raymond Mooney. Semi-supervised clustering by seeding. In *In Proceedings of 19th International Conference on Machine Learning (ICML-2002)*, 2002.

[23] Sugato Basu, Mikhail Bilenko, and Raymond J Mooney. Comparing and unifying search-based and similarity-based approaches to semi-supervised clustering. In *Proceedings of the ICML-2003 workshop on the continuum from labeled to unlabeled data in machine learning and data mining*, pages 42–49. Citeseer, 2003.

[24] Sugato Basu, Mikhail Bilenko, and Raymond J Mooney. A probabilistic framework for semi-supervised clustering. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 59–68. ACM, 2004.

[25] Sugato Basu, Ian Davidson, and Kiri Wagstaff. *Constrained clustering: Advances in algorithms, theory, and applications.* CRC Press, 2008.

[26] Mikhail Belkin and Kaushik Sinha. Polynomial learning of distribution families. In *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, FOCS '10, pages 103–112, Washington, DC, USA, 2010. IEEE Computer Society.

[27] Shai Ben-David. A framework for statistical clustering with constant time approximation algorithms for k-median and k-means clustering. *Machine Learning*, 66(2-3):243–257, 2007.

[28] Shai Ben-David. Computational feasibility of clustering under clusterability assumptions. *CoRR*, abs/1501.00437, 2015.

[29] Shalev Ben-David and Lev Reyzin. Data stability in clustering: A closer look. *Theoretical Computer Science*, 558:51–61, 2014.

[30] Gérard Biau, Luc Devroye, and Gábor Lugosi. On the performance of clustering in hilbert spaces. *Information Theory, IEEE Transactions on*, 54(2):781–790, 2008.

[31] Mikhail Bilenko, Sugato Basu, and Raymond J Mooney. Integrating constraints and metric learning in semi-supervised clustering. In *Proceedings of the twenty-first international conference on Machine learning*, page 11. ACM, 2004.

[32] Avrim Blum. Approximation-stability and perturbation-stability. In *DAGSTUHL Workshop on Analysis of Algorithms Beyond the Worst Case*, 2014.

[33] Siu-On Chan, Ilias Diakonikolas, Rocco A. Servedio, and Xiaorui Sun. Efficient density estimation via piecewise polynomial approximation. In *Proceedings of the Forty-sixth*

Annual ACM Symposium on Theory of Computing, STOC '14, pages 604–613, New York, NY, USA, 2014. ACM.

[34] Sanjoy Dasgupta. Learning mixtures of Gaussians. In *Foundations of Computer Science, 1999. 40th Annual Symposium on*, pages 634–644. IEEE, 1999.

[35] Sanjoy Dasgupta. *The hardness of k-means clustering*. Department of Computer Science and Engineering, University of California, San Diego, 2008.

[36] Ian Davidson and SS Ravi. Agglomerative hierarchical clustering with constraints: Theoretical and empirical results. In *Knowledge Discovery in Databases: PKDD 2005*, pages 59–70. Springer, 2005.

[37] Ayhan Demiriz, Kristin P Bennett, and Mark J Embrechts. Semi-supervised clustering using genetic algorithms. *Artificial neural networks in engineering (ANNIE-99)*, pages 809–814, 1999.

[38] Luc Devroye and Gábor Lugosi. Nonasymptotic universal smoothing factors, kernel complexity and yatracos classes. *The Annals of Statistics*, pages 2626–2637, 1997.

[39] Luc Devroye and Gábor Lugosi. *Combinatorial methods in density estimation*. Springer Series in Statistics. Springer-Verlag, New York, 2001.

[40] Inderjit S Dhillon, Yuqiang Guan, and Brian Kulis. Kernel k-means: spectral clustering and normalized cuts. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 551–556. ACM, 2004.

[41] I. Diakonikolas, G. Kamath, D. M. Kane, J. Li, A. Moitra, and A. Stewart. Robust estimators in high dimensions without the computational intractability. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 655–664, Oct 2016.

[42] Ilias Diakonikolas. Learning Structured Distributions. In Peter Bühlmann, Petros Drineas, Michael Kane, and Mark van der Laan, editors, *Handbook of Big Data*, chapter 15, pages 267–283. Chapman and Hall/CRC, 2016.

[43] Ilias Diakonikolas, Gautam Kamath, Daniel M Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. Robust estimators in high dimensions without the computational intractability. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 655–664. IEEE, 2016.

[44] Ilias Diakonikolas, Gautam Kamath, Daniel M Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. Robustly learning a gaussian: Getting optimal error, efficiently. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2683–2702. SIAM, 2018.

[45] Ilias Diakonikolas, Daniel M Kane, and Alistair Stewart. Learning multivariate log-concave distributions. In *Proceedings of Machine Learning Research*, volume 65 of *COLT'17*, pages 1–17, 2017.

[46] Ilias Diakonikolas, Daniel M Kane, and Alistair Stewart. Statistical query lower bounds for robust estimation of high-dimensional Gaussians and Gaussian mixtures. *arXiv preprint arXiv:1611.03473v2 [cs.LG]*, 2017. To appear in Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS '17).

[47] Brian Eriksson, Gautam Dasarathy, Aarti Singh, and Robert Nowak. Active clustering: Robust and efficient hierarchical clustering using adaptively selected similarities. *arXiv preprint arXiv:1102.3887*, 2011.

[48] Jon Feldman, Rocco A. Servedio, and Ryan O'Donnell. PAC learning axis-aligned mixtures of Gaussians with no separation assumption. In *Proceedings of the 19th Annual Conference on Learning Theory*, COLT'06, pages 20–34, Berlin, Heidelberg, 2006. Springer-Verlag.

[49] Buddhima Gamlath, Sangxia Huang, and Ola Svensson. Semi-supervised algorithms for approximately optimal and accurate clustering. *arXiv preprint arXiv:1803.00926*, 2018.

[50] Michael R Garey and David S Johnson. *Computers and intractability*, volume 29. wh freeman New York, 2002.

[51] Siddharth Gopal and Yiming Yang. Transformation-based probabilistic clustering with supervision.

[52] Mary Inaba, Naoki Katoh, and Hiroshi Imai. Applications of weighted voronoi diagrams and randomization to variance-based k-clustering. In *Proceedings of the tenth annual symposium on Computational geometry*, pages 332–339. ACM, 1994.

[53] Marek Karpinski and Angus Macintyre. Polynomial bounds for VC dimension of sigmoidal and general pfaffian neural networks. *Journal of Computer and System Sciences*, 54(1):169 – 176, 1997.

[54] Michael Kearns, Yishay Mansour, Dana Ron, Ronitt Rubinfeld, Robert E. Schapire, and Linda Sellie. On the learnability of discrete distributions. In *Proceedings of the Twenty-sixth*

Annual ACM Symposium on Theory of Computing, STOC '94, pages 273–282, New York, NY, USA, 1994. ACM.

[55] Taewan Kim and Joydeep Ghosh. Relaxed oracles for semi-supervised clustering. *arXiv preprint arXiv:1711.07433*, 2017.

[56] Taewan Kim and Joydeep Ghosh. Semi-supervised active clustering with weak oracles. *arXiv preprint arXiv:1709.03202*, 2017.

[57] Bo'az Klartag. A central limit theorem for convex sets. *Inventiones mathematicae*, 168(1):91–131, 2007.

[58] Akshay Krishnamurthy, Sivaraman Balakrishnan, Min Xu, and Aarti Singh. Efficient active algorithms for hierarchical clustering. *arXiv preprint arXiv:1206.4672*, 2012.

[59] Brian Kulis, Sugato Basu, Inderjit Dhillon, and Raymond Mooney. Semi-supervised graph clustering: a kernel approach. *Machine learning*, 74(1):1–22, 2009.

[60] Kevin A Lai, Anup B Rao, and Santosh Vempala. Agnostic estimation of mean and covariance. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 665–674. IEEE, 2016.

[61] Martin HC Law, Alexander P Topchy, and Anil K Jain. Model-based clustering with probabilistic constraints. In *SDM*. SIAM, 2005.

[62] David A. Levin, Yuval Peres, and Elizabeth L. Wilmer. *Markov chains and mixing times*. American Mathematical Society, Providence, RI, 2009.

[63] Jerry Li and Ludwig Schmidt. Robust and proper learning for mixtures of gaussians via systems of polynomial inequalities. In Satyen Kale and Ohad Shamir, editors, *Proceedings of the 2017 Conference on Learning Theory*, volume 65 of *Proceedings of Machine Learning Research*, pages 1302–1382, Amsterdam, Netherlands, 07–10 Jul 2017. PMLR.

[64] Nick Littlestone and Manfred Warmuth. Relating data compression and learnability. Technical report, Technical report, University of California, Santa Cruz, 1986.

[65] Stuart P Lloyd. Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, 28(2):129–137, 1982.

[66] Mario Lucic, Matthew Faulkner, Andreas Krause, and Dan Feldman. Training mixture models at scale via coresets. *arXiv preprint arXiv:1703.08110*, 2017.

[67] Meena Mahajan, Prajakta Nimbhorkar, and Kasturi Varadarajan. The planar k-means problem is np-hard. In *WALCOM: Algorithms and Computation*, pages 274–285. Springer, 2009.

[68] Satyaki Mahalanabis and Daniel Stefankovic. Density estimation in linear time. In *21st Annual Conference on Learning Theory - COLT 2008, Helsinki, Finland, July 9-12, 2008*, pages 503–512, 2008.

[69] Andreas Maurer and Massimiliano Pontil. k-dimensional coding schemes in hilbert spaces. *Information Theory, IEEE Transactions on*, 56(11):5839–5846, 2010.

[70] Arya Mazumdar and Soumyabrata Pal. Semisupervised clustering, and-queries and locally encodable source coding. In *Advances in Neural Information Processing Systems*, pages 6492–6502, 2017.

[71] Arya Mazumdar and Barna Saha. Clustering with noisy queries. In *Advances in Neural Information Processing Systems*, pages 5790–5801, 2017.

[72] Arya Mazumdar and Barna Saha. Query complexity of clustering with side information. In *Advances in Neural Information Processing Systems*, pages 4685–4696, 2017.

[73] Colin McDiarmid. *Concentration*, pages 195–248. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998. available at http://www.stats.ox.ac.uk/people/academic_staff/colin_mcdiarmid/?a=4139.

[74] Vincent Michel, Alexandre Gramfort, Gaël Varoquaux, Evelyn Eger, Christine Keribin, and Bertrand Thirion. A supervised clustering approach for fmri-based inference of brain states. *Pattern Recognition*, 45(6):2041–2049, 2012.

[75] Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, New York, NY, USA, 2005.

[76] Ankur Moitra and Gregory Valiant. Settling the polynomial learnability of mixtures of Gaussians. In *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, FOCS '10, pages 93–102, Washington, DC, USA, 2010. IEEE Computer Society.

[77] Shay Moran and Amir Yehudayoff. Sample compression schemes for vc classes. *Journal of the ACM (JACM)*, 63(3):21, 2016.

[78] David Pollard. *Convergence of stochastic processes*. David Pollard, 1984.

[79] Ananda Theertha Suresh, Alon Orlitsky, Jayadev Acharya, and Ashkan Jafarpour. Near-optimal-sample estimators for spherical Gaussian mixtures. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 1395–1403. Curran Associates, Inc., 2014.

[80] Wei Tang, Hui Xiong, Shi Zhong, and Jie Wu. Enhancing semi-supervised clustering: a feature projection perspective. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 707–716. ACM, 2007.

[81] Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[82] V. N. Vapnik and A. Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability & Its Applications*, 16(2):264–280, 1971.

[83] Andrea Vattani. The hardness of k-means clustering in the plane. *Manuscript, accessible at http://cseweb. ucsd. edu/avattani/papers/kmeans_hardness. pdf*, 617, 2009.

[84] Konstantin Voevodski, Maria-Florina Balcan, Heiko Röglin, Shang-Hua Teng, and Yu Xia. Active clustering of biological sequences. *The Journal of Machine Learning Research*, 13(1):203–225, 2012.

[85] Ulrike Von Luxburg and Shai Ben-David. Towards a statistical theory of clustering. In *Pascal workshop on statistics and optimization of clustering*, pages 20–26, 2005.

[86] Kiri Wagstaff, Claire Cardie, Seth Rogers, Stefan Schrödl, et al. Constrained k-means clustering with background knowledge. In *ICML*, volume 1, pages 577–584, 2001.

[87] Eric P Xing, Michael I Jordan, Stuart Russell, and Andrew Y Ng. Distance metric learning with application to clustering with side-information. In *Advances in neural information processing systems*, pages 505–512, 2002.