# Faithful Representations and Moments of Satisfaction: Probabilistic Methods In Learning and Logic

Thesis submitted for the degree
"Doctor of Philosophy"

Lidror Troyansky

Submitted to the Senate of the Hebrew University in the year 1998

This work was carried out under the supervision of

Prof. Naftali Tishby.

To my wife, Ma'ayan, and my daughter, Shira.

# Acknowledgments

Special thanks are due to:

- Prof. Naftali Tishby for his help and guidance in carrying out this study, for the many fascinating discussions we had, and for the immense body of knowledge that I have absorbed from him during my studies.
- Prof. Scott Kirkpatrick, Prof. Bart Selman, Dr. Remi Monasson, Dr. Riccardo Zecchina, Daniel Gill and Dr. Israel Nelken for fruitful collaboration in various parts of this research.
- The Ph.D. students of machine learning group: Itay Gat, Shai Fine, Omri Mann, Elad Schneidman, Noam Slonim and Golan Yona and to Dr. Shlomo Dubnov and Dr. Ran El-Yaniv for their friendship, help and support during this study.
- The Ms.C. students of the lab, Gill Bejerano, Adi Schreibman and Michael Ne'eman for bringing a new spirit to the lab.
- My wife, Ma'ayan and my daughter, Shira, for decorating my life and for their patience.
- My parents and the rest of my family and my good friends, for their love, help and support.
- The Clore Foundation, for providing me a generous financial support during my studies.

# Contents

# List of Figures

## Abstract

The study described in this thesis is concerned with two related problems that stem from research in the field of artificial intelligence: supervised learning from examples and the problem of constraint satisfaction of logical formulae. The research is multi-disciplinary, and combines methods and techniques from computational learning theory, probability, and statistical physics. The research also contains applications of neural network methods in order to gain insight into sensory information processing in a biological system.

The first part of the thesis contains an introduction and a general survey of the field of computational learning from examples. The second part of the thesis is concerned with the problem of the definition and construction of adequate representations for learning: in chapter 4 we introduce a method for construction of faithful and balanced binary representations of input instances and hypotheses. We consider the Hamming distance between the constructed representations of any two hypotheses, and show that this distance approximates the probability of their disagreement with respect to the labels of input instances (i.e., the relative generalization error of one hypothesis w.r.t. the other). Using a "dual learning problem" setting, we also show that the Hamming distance between the representation of any two input instances approximates the probability that a random hypothesis will not assign the same label to the two inputs. The representations are balanced in the sense that all the coordinates in the representation are statistically of equal weight.

Chapter 5 describes two learning methods that make use of faithful and balanced representations. The first applies a "local" algorithm that uses the correlations between the representations of the target hypotheses and the representations of a pre-defined set of hypotheses.[1] The second employs least-mean-square approximators that attempt to approximate a separating hyperplane between the positive and the negative examples. Results of simulations of these algorithms are briefly described.

The third part of the thesis is concerned with the problem of constraint satisfaction, both in the context of supervised learning from examples and in the general context of satisfying a Boolean expression. Chapter 6 discusses the relation between supervised learning and constraint satisfaction problems, and analyzes the dual problem of CNF learning as a prototype. The average sample complexity of this problem is analyzed and a learning algorithm, based on a new "phase-space annealing" method is introduced. This algorithm achieves good performance in solving typical k-CNF formulae, a problem of tremendous practical importance.

Chapter 7 deals with the general problem of determining the satisfiability of k-CNF formulae (the k-SAT problem) and the typical computational complexity of finding a satisfying assignment to a k-CNF formula. In the analysis of this problem we employed several probabilistic methods inspired by statistical mechanics of disordered materials and neural networks with a generalized Hebbian rule.

The probability of satisfiability of a randomly chosen k-CNF formula is studied asymptoti-


[1]This setting is inspired by a basic biological learning mechanism, known as Hebb's rule, and we therefore refer to this setting as the "Hebb-like algorithm".

cally, as the ratio between the number of conjunctive clauses, $M$, and the number of variables, $N$, denoted by $\alpha = M/N$, is fixed, and both $M$ and $N$ tend to infinity. It is exponentially close to one when $\alpha$ is small, and exponentially close to zero when $\alpha$ is large. Further, the transition between the range of "small" $\alpha$ and "large" $\alpha$ is very sharp and occurs around a certain threshold value $\alpha_c$. It turns out that the resolution of a typical k-SAT instance, $k \geq 3$, is intractable only in the immediate vicinity of the threshold. Understanding the threshold phenomenon and the origin of intractability is therefore of great importance. Several original contributions to the understanding of this problem are described: a method for obtaining the shape of the threshold is introduced and a formula for the shape of the threshold in the large-k limit is derived. Section 7.4 describes a statistical mechanics formulation for the satisfiability problem, which is used in order to characterize the properties of typical large k-CNF formulae. An order parameter that measures the average overlap between satisfying assignments is derived, using the second moment of the number of satisfying assignments, and is used in order the predict the value of the threshold. The results are in good agreement with experimental results for $k \geq 3$.

In order to gain a better understanding of the origin of computational complexity, we [2] introduced a model that smoothly interpolates between the always-tractable 2-SAT problem and the intractable (in the worst case) 3-SAT problem: the 2+p-SAT model, in which an ensemble of CNF formulae with fraction p of 3–clauses and fraction $1-p$ of 2–clauses is constructed. We found that, up to a critical $p_0$, statistical mechanics characteristics of the problem are similar to those of 2-SAT, while above this critical $p_0$, the problem has the same characteristics as the 3-SAT problem. Simulations results show, indeed, that at the critical $\alpha$, typical 2+p-CNF formulae are tractable up to $p_0$ and are intractable above $p_0$. The analysis suggests a possible explanation for the origin of intractability.

The last part of the thesis (chapter 8) employs learning methods of artificial neural-networks to address the biological question: the estimation of the direction of an unfamiliar sound source based on monoaural cues. [3]: The major cues for sound localization are Interaural Time Differences (ITD) and Interaural Intensity Differences (IID) between the two ears. However, these cues cannot be used for determining the elevation of sound sources located on the medial saggital plane. The pinnae, head, and torso modify the spectrum of the sound signal; this distortion, which is direction-dependent, is quantified by measurement of the transfer function from free-field to ear drum, known as the *Head Related Transfer Function* (HRTF). In order to extract localization information from the monaural spectrum, the auditory system must be able to distinguish between the source spectrum and the HRTF. In this chapter we present a neural network model motivated by biological considerations for the hardest localization problem, namely elevational allocation of unfamiliar natural sound signals, based solely on monoaural direction-dependent spectral information, using a real HRTF data. The resulting network was able to determine the elevation of an unfamiliar sound to within $\pm 3.5°$ in all of the test cases.

[2]This part of the study was done in collaboration with R. Zecchina, R. Monasson, S. Kirkpatrick and B. Selman

[3]This part of the study was done in collaboration with Daniel Gill and Israel Nelken

# Part I

# General Background

# Chapter 1

# Introduction

The ability of biological systems to learn is of paramount importance to the survival of species in complicated and ever changing environments. Learning skills, together with a related, more elusive capability of "creativity," are the cornerstones of the unique evolutionary niche of Homo Sapience: the intelligence niche. It is the overwhelming potential of this niche that allows me to write this thesis and allows you to read it, rather than spending our time on basic survival actions, such as hunting, collecting food, and escaping predators.

The process of learning has many different aspects: a parrot who learns to utter a phrase, a little child who learns to speak, a mathematician who understands a sophisticated proof, an aplysia that was conditioned to perform a gill-withdrawal reflex and a neural-network based computer program that can classify patterns – have all been involved in a learning process of some kind. However, the nature of the learning process in those cases is very different. It is therefore difficult to find a precise definition of learning. A die hard behaviorist would say that learning occurred only if it is accompanied by a measurable change in behavior, and that learning should therefore be defined as: *The modification of behavior through practice, training or experience* (from the Random House Dictionary) However, this definition precludes the gain of "knowledge for the sake of knowledge," which was admired by scholars through the ages, from being "true learning." A less restrictive definition is suggested by the Webster dictionary: *To gain knowledge or understanding of, or skill in, by study, instruction, or experience (learn a trade)* Still, this definition relies on the problematic notions of "knowledge" and "understanding."

Moreover, the introduction of computers adds a new perspective and new needs for the study of learning. Ever since the early days of computers, it was clear that adding learning capabilities to machines will greatly enhance their performance. The notion of artificial intelligence (AI) is strongly connected with machine learning: since the ability to learn is an inalienable component of human intelligence, one cannot have real "artificial intelligence" without giving the machine an ability to learn, to adapt, and to modify its behavior. However, the immense difficulties that the early researcher confronted in trying to endow learning capabilities to computers, cause a shift towards paradigms which seemed at a time more suitable for computers, namely, the rule-based systems; in these systems the decisions are governed by a set of pre-defined rules and logical conditions. Rule-based decisions are indeed very useful in cases where there is a relatively simple set of underlying rules which can be pre-defined and can lead to quick and optimal decisions. However, in most real life situations it is very hard (if not impossible) to pre-define such a set of rules: one cannot hope to anticipate every set of circumstances that a program might encounter, and even if we could, and we built in all the appropriate responses, the changing world would soon make the resulting frozen knowledge base obsolete. Thus, flexible, multi-purpose learning is essential for programs that deal with the real world.

The impressive progress in the theoretical and the practical research in machine learning that was achieved during the last three decades, has made machine learning one of the pillars of mainstream AI research. Machine learning now stands on a firm mathematical base and a mature machine learning technology finds its use in many commercial pieces of software for tasks such as optical character recognition (OCR)[93] [65], system control [126] [80], speech recognition, pattern recognition, industrial process optimization and control, medical expert systems [69] [9]and many others. In addition, the theoretical study of learning processes enhances our understanding of learning in biological systems and contributes to the de-mystification of higher cognitive functions.

## 1.1 General organization of the thesis and main results

This thesis deals with supervised learning of concepts from examples ("inductive learning"), as explained in section 1.2.2, and with the related problem of constraint satisfaction, described in section 6.1. Since contemporary learning theory is rather diverse, and since several new methods and concepts are related to our results, the introduction to the thesis is rather long. The first three chapters are devoted to preliminaries and to a general overview of the field. Chapter 1 describes the general notion and the basic paradigms of supervised learning, including some mathematical essentials that will be needed throughout the thesis.

Chapter 2 describes formal models of concept learning and function learning. In particular, the chapter introduces the notion of Probably Approximately Correct (PAC) learnability, and discusses the conditions required for learning and generalization. The relation between learnability and uniform convergence of frequencies to probabilities – one of the most important results of computational learning theory – is discussed in section 2.3, and necessary and sufficient conditions for uniform convergence are discussed in sections 2.3.1 and 2.3.2 The dual learning problem, where one attempts to infer an input from a sample of hypotheses is discussed in section 2.4, and an important class of learning problems, linearly separable problems, are introduced and discussed in section 2.5. Finally, learning of real-valued functions is briefly discussed in section 2.6.

Chapter 3 is concerned with "universal" learning algorithms – i.e., problem independent learning methods. The subject is introduced in section 3.1. A rather new and general family of classifiers, the support vector machines, are discussed in section 3.2. Section 3.3. describes the successful method of "boosting," which combines classifiers whose performance is only slightly better than random into a "strong" classifier, with low generalization error. Section

3.4. describes decision trees – another successful family of classifiers. A method for learning Boolean functions using the Fourier transform is described in section 3.5. Radial-Basis functions and Gaussian-Processes classifiers are described in sections 3.6-3.7, and multiple task learning, a method which can transfer an inductive bias from one learning problem to another, is briefly discussed in section 3.8.

Original results are described in chapters 4-8. In chapter 4 we introduce a method for construction of faithful and balanced binary representations of input instances and hypotheses. We consider the Hamming distance between the constructed representations of any two hypotheses, and show that this distance approximates the probability of their disagreement with respect to the labels of input instances (i.e., the relative generalization error of one hypothesis w.r.t. the other). Using a "dual learning problem" setting, we also show that the Hamming distance between the representation of any two input instances approximates the probability that a random hypothesis will not assign the same label to the two inputs. The representations are balanced in the sense that all the coordinates in the representation are statistically of equal weight. Furthermore, the fact that the representations are binary allows efficient evaluation of distances and correlations between hypotheses and between input instances.

Chapter 5 describes two learning methods that make use of faithful and balanced representations. The first applies a "local" algorithm that uses the correlations between the representations of the target hypotheses and the representations of a pre-defined set of hypotheses. This setting is inspired by a basic biological learning mechanism, known as Hebb's rule, and we therefore refer to this setting as the "Hebb-like algorithm". The second employs least-mean-square approximators that attempt to approximate a separating hyperplane between the positive and the negative examples. Results of simulations of these algorithms are briefly described.

The third part of the thesis is concerned with the problem of constraint satisfaction, both in the context of supervised learning from examples and in the general context of satisfying a Boolean expression. Chapter 6 discusses the relation between supervised learning and constraint satisfaction problems, and analyzes the dual problem of CNF learning as a prototype. The average sample complexity of this problem is analyzed and a novel learning algorithm, based on a new "phase-space annealing" method is introduced. This algorithm achieves good performance in solving typical k-CNF formulae, a problem of tremendous practical importance.

Chapter 7 deals with the general problem of determining the satisfiability of k-CNF formulae (the k-SAT problem) and the typical computational complexity of finding a satisfying assignment to a k-CNF formula. In the analysis of this problem we employed several probabilistic methods inspired by statistical mechanics of disordered materials and neural networks with a generalized Hebbian rule. The k-SAT problem, for any $k > 2$, is known to belong to the NP-Complete class. This, however, does not say much about its *typical computational complexity*, i.e., the number of computational steps needed for the resolution of a typical instance.

The probability of satisfiability of a randomly chosen k-CNF formula is studied asymptotically, as the ratio between the number of conjunctive clauses, $M$, and the number of variables, $N$, denoted by $\alpha = M/N$, is fixed, and both $M$ and $N$ tend to infinity. It is exponentially close to one when $\alpha$ is small, and exponentially close to zero when $\alpha$ is large. Further, the transition between the range of "small" $\alpha$ and "large" $\alpha$ is very sharp and occurs around a certain threshold value $\alpha_c$. It turns out that the resolution of a typical k-SAT instance, $k \geq 3$, is intractable only in the immediate vicinity of the threshold. Understanding the threshold phenomenon and the origin of intractability is therefore of great importance. Several original contributions to the understanding of this problem are described: a method for obtaining the shape of the threshold is introduced and a formula for the shape of the threshold in the large-k limit is derived. The results are compared with a re-scaled version of the shape of the threshold found in numerical simulations, and they are found to be in very good agreement. Section 7.4 describes a statistical mechanics formulation for the satisfiability problem, which is used in order to characterize the properties of typical large k-CNF formulae. An order parameter that measures the average overlap between satisfying assignments is derived, using the second moment of the number of satisfying assignments, and its rate of change with respect to $\alpha$ is used in order the predict the value of the threshold. The results are in good agreement with experimental results for $k \geq 3$.

In order to gain a better understanding of the origin of computational complexity, we [1] introduced a model that smoothly interpolates between the always-tractable 2-SAT problem and the intractable (in the worst case) 3-SAT problem: the 2+p-SAT model, in which an ensemble of CNF formulae with fraction p of 3–clauses and fraction $1 - p$ of 2–clauses is constructed. We found that, up to a critical $p_0$, statistical mechanics characteristics of the problem are similar to those of 2-SAT, while above this critical $p_0$, the problem has the same characteristics as the 3-SAT problem. Simulations results show, indeed, that at the critical $\alpha$, typical 2+p-CNF formulae are tractable up to $p_0$ and are intractable above $p_0$. The analysis suggests a possible explanation for the origin of intractability: for $p > p_0$ and $\alpha > \alpha_c$ all the solutions ("ground states") of a typical formula contain a "backbone": a set of variables whose value have to be same in the solutions. The existence of such a backbone poses a difficult problem for various heuristics for solving k-SAT problems.

The last part of the thesis (chapter 8) employs learning methods of artificial neural-networks to address the biological question: the estimation of the direction of an unfamiliar sound source based on monoaural cues. [2]: it is known that many species of animals can identify the direction of arrival of sounds. The major cues for sound localization are Interaural Time Differences (ITD) and Interaural Intensity Differences (IID) between the two ears. However, these cues have approximately the same values for locations on "cones of confusion" and in particular cannot be used for determining the elevation of sound sources located on the medial saggital plane. The pinnae, head, and torso modify the spectrum of the sound signal; this distortion, which is direction-dependent, is quantified by measurement of the transfer function from free-field to ear drum, known as the *Head Related Transfer Function* (HRTF). It is accepted today that the spectral features introduced by the HRTFs affect sound localization and may serve as cues for disambiguation of locations on the cones of

[1] This part of the study was done in collaboration with R. Zecchina, R. Monasson, S. Kirkpatrick and B. Selman

[2] This part of the study was done in collaboration with Daniel Gill and Israel Nelken

confusion. The use of HRTFs for sound localization poses a challenge to the auditory system: spectral features of a sound at the ear drum can be either properties of the sound source itself, or properties of the HRTF. Thus, in order to extract localization information from the monaural spectrum, the auditory system must be able to distinguish between the source spectrum and the HRTF. In this chapter we present a neural network model motivated by biological considerations for the hardest localization problem, namely elevational allocation of unfamiliar natural sound signals, based solely on monoaural direction-dependent spectral information, using a real HRTF data. The resulting network was able to determine the elevation of an unfamiliar sound to within $\pm 3.5°$ in all of the test cases.

During my Ph.D. studies I was involved, together with my advisor, in another research, which is not related to the main course of the research described in this thesis: analysis of the quantum evaluation of the determinant and the permanent of a matrix. While the determinant of a matrix can be evaluated in time that is polynomial in the size of the matrix, all current algorithms for the evaluation of the permanent of a real matrix have exponential time complexity and are known to be in the in the class $P^{\#P}$. Permanents and determinants of matrices play a basic role in quantum statistical mechanics of identical bosons and fermions, as the only possible many particle states made of single particle wave functions. We study a novel many-particle quantum-computational model in which an observable operator can be constructed, in polynomial-time complexity, to yield the determinant or the permanent of an arbitrary matrix as its expectation value. Both quantities are estimated, in this model, by quantum mechanics systems in a polynomial time, using fermions and bosons respectively. It turns out that the *variance* of the measurements in a "noise-free case" is zero for the determinant and exponential (in the rank of the matrix) for the permanent. The intrinsic measurement variance is therefore the quantum mechanical correspondence to the computational complexity gap. The results of this study appear in the appendix.

The rest of the following chapter is organized as follows: section 1.2.1 briefly describes several learning paradigms from the AI literature. Section 1.2.2 describes the basic notion of the supervised learning, which is the main concern of this work. Section 1.3 describes some probabilistic formulation of supervised learning. In section 1.4 some mathematical essentials, which will be needed throughout the thesis are defined, and conditions for learnability are described.

## 1.2 Preliminaries

### 1.2.1 Learning paradigms in artificial intelligence

The problem of machine learning has various aspects, and researchers in the field of artificial intelligence have developed several methods and techniques in order to construct machines that learn[3]. We bring here, in a nutshell, some examples for such paradigms:

[3] Since the definition of learning is rather vague, there is no general agreement regarding what can be considered as real "machine learning," and naming some of these paradigms as "learning methods" may be controversial

- Memorization: the simplest kind of learning is just information storage. this kind of "learning" can be improved if information is stored in an organized manner, which allows queries to be answered efficiently. A major step in making memorization more "human" is *content addressable,* or *associative* memories, in which the information can be retrieved based on partial or "noisy" version of it.

- Parameter adjustment: problem-solving programs use evaluation functions (or heuristic functions) in order to evaluate the merit of various operations. The parameters of this function can be learned, or adjusted, as the system gains more experience.

- *Rote learning:* In "rote learning," the computational performance of a program is improved in time, by storing information that was obtained during the computational steps required for solving one problem and using it for the solution of other problems.

- In *learning by taking advice* the heuristic function can be altered by an external advice (e.g. "get control over the center of the board" in a chess-playing program)

- In learning by *chunking* sequences of actions, which the learning system found advantageous in previous cases, are stored in order to be used later on.

- In *explanation-based learning* the system attempts to learn from a single example $x$ by explaining why $x$ is an example to the target concept.

- *Discovery* is a restricted form of learning in which one entity acquires knowledge without the help of a teacher. Three main types of discovery are *theory driven discovery*, in which the system attempts to discover new and interesting concepts within the framework of a formal system (such as number theory), *data driven discovery*, in which the system attempts to find interesting connections and rules in the data supplied, and *clustering*, in which the system attempts to find sub-groups and structures in the data, based on some similarity or distance criteria.

- In *analogy-based learning* the system attempts to solve new problems by making analogies to previously solved problems.

### 1.2.2 The problem of supervised learning from examples

We distinguish between supervised and un-supervised learning: in supervised learning there is an interaction between the learner and a *teacher*. The teacher gives the learner feedback on its actions, and the learner can adjust its behavior according to this feedback: to alter a behavior that yields negative feedback and to promote behavior that yields positive feedback. An effective "teacher" for life-forms is their environment, and even simple life-forms can alter behavior based on environmental cues.

The simplest type of supervised learning is *classical conditioning*. When a cue ("conditioned stimulus") is supplied a sufficient number of times together with a meaningful stimulus, such as food or an electric shock ("unconditioned stimulus"), the animal learns to associate the stimulus with the cue, and behavior that corresponds to the meaningful stimulus will appear whenever the animal receives the cue. This type of learning was demonstrated by the celebrated experiments of Ivan Pavlov at the turn of the century [132], where dogs were exposed to food after hearing a ring of a bell: after a conditioning period, the dogs responded with excessive salivation to the ring of the bell even in the absence of food. In general, animals are capable of producing complex behavioral changes in order to get positive feedback, or to avoid negative feedback, from their environment (e.g., to learn to find their way in a complex maze in order to find food, to go to work in order to get the pay-check in the end of the month, and to write a thesis in order to get a doctoral degree).

In the context of machine-learning, supervised learning from examples (also known as *inductive inference*) is one of the most prevalent paradigms. The basic formal paradigm for learning from examples is due to E.M. Gold (1967)[64]. Within this paradigm, a learning algorithm is presented with an infinite stream of examples; an example is an element from the domain of interest, with a label indicating its membership in the set. After observing each example, the algorithm conjectures a hypothesis for the unknown set. The algorithm is said to *converge in the limit* if its hypothesis converges to the unknown set after finitely many examples. Such convergence is required over all admissible streams of examples.

To illustrate this learning paradigm, consider the case of learning a rectangle from $R^2$ which was chosen from a pre-defined finite set of rectangles. The unknown set is the set of all points in $R^2$ which reside inside the rectangle. An example would be a point from $R^2$, together with a label which would be positive if the point resides inside the rectangle, and negative otherwise. The algorithm might suggest any rectangle from the pre-defined set as an hypothesis. If, using a finite number of examples, the algorithm will conjecture the correct rectangle – we say that the algorithm converged to the right answer. Once we know the correct rectangle, there is no need for labels: given the coordinates of any point in $R^2$, the algorithm can decide with absolute certainty whether it resides inside or outside the rectangle. The ability to predict the label of a novel input, using a given sample of labeled input instances is called *the generalization ability*.

However, Gold's paradigm does not take into account some important issues, such as: what if we can only find, using any finite sample, a hypothesis that is a good approximation to the correct rectangle? What is the number of labeled examples which are needed, and on which parameters of the learning problem does this number depend? Is the learning algorithm computationally efficient? As we will see, these questions and others are answered by probabilistic paradigms of machine learning. But before we address those paradigms, we will first consider the problem of generalization.

### 1.2.3 The problem of generalization

When dealing with inputs from the real world, the chances to encounter exactly the same input twice are negligibly small: the physical signal which corresponds to a given spoken word has an enormous number of variations, the same object produces different visual stimulus in different angles and in different light conditions, and even a simple tactile stimuli can be produced with an infinite number of intensities. Learning will be almost useless if it could only deal with inputs which have been previously encountered by the learner. One of the most important characteristics of a learning system, whether biological or artificial, is therefore its ability to *generalize:* to classify correctly input instances that have never been encountered before, based on a set of labeled samples.

On the cognitive level, generalization is a most effective scheme to digest information and to obtain prediction about future events. For example, if we see enough examples of "dogs," we can learn the concept of "dog." Then, once we have seen Rexi, we do not need anyone to tell us that it is a dog, and we also have a prior knowledge about its possible behavior. As such, generalization is, perhaps, the most important ingredient of human intelligence, and many IQ tests (e.g., series completion) are devoted to check the examinee's generalization capabilities.

From the machine learning (or inductive inference) standpoint, good generalization capabilities are the basic demand from a learning algorithm. In chapter 2 we will discuss when, and under what conditions good generalization capabilities can be achieved.

## 1.3 Probabilistic formulation of supervised learning

If we are willing to accept a less-than-certain generalization performance, learning problems may become considerably simpler. Consider the problem of learning an axis-aligned rectangle in $R^2$. (Fig. 1.1) The input instances are points that are drawn uniformly from a bounded subset of $R^2$ that contains the rectangle. Given a sample of $M$ labeled points (where the label of the point is positive if the point resides inside the rectangle, and negative otherwise) we can find a rectangle that is consistent with the sample (i.e., all the positive examples reside inside it and all the negative examples reside outside) using the coordinates of the extreme positive examples in each dimension. But as long as the sample is finite, there would be infinitely many consistent rectangles, and without sufficient prior knowledge the chance to find the "true" rectangle is zero.

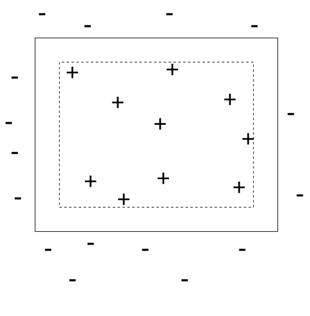

Figure 1.1: Learning of a rectangle in $R^2$: the learner attempt to approximate the target rectangle (solid line) using a consistent rectangle (dashed line).

However, if the sample is large enough, every consistent rectangle should become close enough to the correct rectangle, so that we can predict, with a high confidence level, the label of any point in $R^2$. If the probability of error is small enough, this may suffice for all practical purposes. Probabilistic paradigms can therefore provide efficient learning methods which yield "good enough" results. Furthermore, the analysis of such methods can provide quantitative measures for the performance of learning algorithms. In this section several basic probabilistic paradigms will be described.

### 1.3.1 The PAC Model

The most influential paradigm in formal learning theory is the "Probably Approximately Correct" (PAC) paradigm which was introduced by L. G. Valiant in his seminal paper "A Theory of the Learnable" [175]. Valiant's PAC model is a mathematical model for learning concepts from examples. A concept according to Valiant is a set of instances (e.g., the set of all dogs), each defined by a set of features (e.g., height, weights, number of legs, etc,) Given a concept, there exists a *rule* according to which instances can be categorized as *belonging* to the concept, or *not belonging* to the concept. A *learning algorithm* is given a sample of independently chosen examples, together with their labels. The learning algorithm must output a rule that categorizes well *novel* instances, namely instances not contained in the sample. Valiant introduced several restrictions and assumptions, as well as criteria for the performance of the learning algorithm, which make this model extremely useful:

- The class of concepts, or rules, $\mathcal{H}$, from which the need-to-be learned rule is taken is well defined and known to the learning algorithm. (for example, axis-aligned rectangles, neural-networks with a certain architecture, 3-CNF Boolean formulae of $M$ clauses over $N$ variables, etc.)

- There is an arbitrary *probability distribution* $\mathcal{D}$ on the instances, un-known to the learning algorithm, which is fixed in the following sense: the distribution according to which the instances in the sample are chosen is the same as the distribution by which new instances, which the algorithm must categorize, are chosen. Due to this assumption the PAC learning model is also known as the *distribution free* learning model.

- The algorithm does not have to output a *perfect* rule, which give the correct label to *every* instance: it is allowed a small probability of error while classifying novel instances, where the error probability is taken with respect to the distribution on the instances. The probability of error is known as the *generalization error* and is usually denoted by $\epsilon$.
  This measure of performance is, indeed, the relevant measure (provided that the distribution remains fixed) since we are interested in a good classification of input instances and not in the rule itself. [4]

- Since the sample is chosen randomly, there is always a chance that the learning may receive an un-representative sample, that would lead to a rule with a large generalization error. We thus allow the algorithm to be only *probably* approximately correct, namely, it is allowed to fail to give a good approximation with some probability, where the

[4]from a psychological perspective, such a measure can be viewed as a behavioristic measure: only measurable performance counts

probability is taken over the choice of the random sample. The probability of failure is usually denoted by $\delta$.

- The learning algorithm is required to be efficient, both in terms of the sample size it needs (its *sample complexity*) and in terms of its running time (its *computational complexity*). The notion and definition of efficiency is taken from complexity theory: a learning algorithm is said to be *efficient* if both its sample complexity and its computational complexity are polynomial in the relevant parameters of the problem.

### 1.3.2 Some extensions of the PAC model

In a real-life learning problem, the setting may be quite different than the basic setting described above. Several modifications and extensions of Valiant's original model were therefore suggested. Two modifications that are relevant to the topic of this thesis are:

- *Learning under a specific distribution:* the restriction that the distribution according to which the input instances are chosen is arbitrary might make the learning task very hard. However, while dealing with a specific learning problem, it suffices to require that the algorithm perform well with regard to the specific distribution at hand. Analysis of the complexity of the task may also be easier if a specific distribution (e.g. the uniform distribution) is assumed.

- *Noisy examples:* real-world data is usually conveyed with noise. Several models of learning in the presence of noise were suggested and investigated. The most basic noise model is the *classification noise* model, in which it is assumed that with some probability each random example is labeled incorrectly (i.e. positive examples are labeled negative, or vice versa).

### 1.3.3 Bayesian learning

Another probabilistic paradigm for supervised learning is the *Bayesian Paradigm*. In this paradigm, a target hypothesis is drawn at random according to a probability distribution $\mathcal{P}$ from the set of all possible hypotheses $\mathcal{H}$. Given a sample of $m$ labeled examples: $S^{(m)} = ((x_1, y_1), (x_2, y_2) \ldots (x_m, y_m))$ where the x's are the input instances and the y's are their labels, the posterior probability of a hypothesis $h \in \mathcal{H}$ is, by Bayes law[7]:

$$P_m\left(h|S^m\right) = \frac{P_0(h)}{P_0(S^m)} P\left(S^{(m)}|h\right). \tag{1.1}$$

Since the sample is drawn independently of $h$, the posterior probability depends only on the set of labels $((y_1, \ldots y_m))$. In the absence of labeling noise,

$$P\left((y_1, \ldots y_m)|h\right) = \begin{cases} 1 & \text{if } h(x_i) = y_i, \text{ for } i = 1..m \\ 0 & \text{otherwise} \end{cases} \tag{1.2}$$

Therefore, only hypotheses that are consistent with the sample labels, and whose prior was positive, would have a positive posterior. The set of such hypotheses is known as the *version space*, and will be denoted by $\mathcal{H}_m$:

$$\mathcal{H}_m\left(S^{(m)}, h\right) = \{h \in \mathcal{H} | h(x_i) = y_i,\ i = 1...m\}\,. \tag{1.3}$$

Given a novel input instance, a *Bayes-optimal* method, namely, one that minimizes the probability of incorrect prediction, is to use a "weighted majority" of the predictions of the hypotheses in $\mathcal{H}_m$, where the weight of each hypothesis is its posterior probability [43] [99]. If the labels are denoted by +1 for a positive example and -1 for a negative example, then the label that the Bayes algorithm assigns to an input instance $x$ is the sign of the expected value of its label in the version space:

$$\begin{aligned} h^m_{_B}(x) &= \mathrm{sgn}\left(\sum_{h\in\mathcal{H}_m} P_m(h)\cdot h(x)\right) \quad \text{(discrete case)} && (1.4)\\ &= \mathrm{sgn}\left(\int_{h\in\mathcal{H}_m} P_m(h)\cdot h(x) d\mathcal{P}(h)\right) \quad \text{(continues case)} && (1.5)\\ & && (1.6)\end{aligned}$$

An obvious drawback of the Bayes algorithm is that it requires knowledge of the prior over the hypotheses space $\mathcal{P}(h)$. However, the algorithm is not very sensitive to this prior if the sample is large enough [71]. The fact that the output hypothesis, being a sign of a linear superposition of hypotheses in $\mathcal{H}$, may not by itself belong to $\mathcal{H}$ may also be considered as a drawback.

### 1.3.4 Statistical physics formulation of supervised learning

Statistical physics explores systems that are composed of a large number of similar components. Relevant features of such systems can often be described in terms of a small number of global parameters (e.g., the physical properties of an ideal gas can be described in terms of its pressure and its temperature). Statistical Physics analysis is usually based on minimization of a *free energy function.* The free energy function includes an *energy term* and an *entropy term,* which is a function of the number of possible configurations of the system and their probabilities. If the system is large enough, its behavior is dominated by configurations that lie in the immediate vicinity of the minima of the free-energy function. Statistical physics analysis can reveal the emergence of collective behavior of the system's components, and to predict sharp transitions in the values of the global parameter (phase transitions). Example to phase transitions are the transitions from solid to liquid and from solid to gas.

In the context of supervised learning, each hypothesis is considered as a configuration of the system. One can define an "energy" $E(h)$ of a hypothesis $h$ as a function of the difference between the hypothesis outputs and the desired outputs for a given set of inputs. Zero energy means that the hypothesis yield the desired output for all the inputs. One can assign a probabilistic weight for the hypotheses using an exponential function of their energy:

$$P(h) \propto e^{-\beta E(h)}. \tag{1.7}$$

where $\beta$ is an arbitrary constant also known as "the inverse temperature." This distribution function is known as the *Gibbs distribution*.

The Gibbs distribution therefore assigns priors to the hypotheses based on the degree of their consistency with the desired outputs. If hypotheses are drawn at random according to Gibbs distribution, the probability for a certain "energy" is determined by the number of hypotheses with this energy, $\mathcal{N}(E)$ together with the appropriate probabilistic weight. For a uniform distribution, this can be written as:

$$P(E) \propto e^{\log \mathcal{N}(E) - \beta E} = e^{-(\beta E - \log \mathcal{N}(E))} \tag{1.8}$$

If the number of hypotheses is large enough, $P(E)$ is non-negligible only in the immediate vicinity of the minimum of $F = \beta E - \log \mathcal{N}(E)$, and the system can be characterized by this minimum. [5]

Statistical physics methods have been successfully employed for the analysis of large neural networks [167] [66] for improved bounds on the error probability in a Bayesian setting [72], and as a motivation for learning algorithms such as the "Boltzmann Machine" [2], the "Gibbs Machine" [121], and the "Helmholtz Machine" [38].

## 1.4 Uniform convergence of frequencies to probabilities

Chernoff and Hoeffding bounds assure us that, given a sequence of $m$ independent trials, the relative frequency of occurrence of an event $a$, $\nu^m(a) = \frac{n(a)}{m}$ converges (in probability) to the probability of this event, $p(a)$, as $m$ tends to infinity. (This is essentially Bernoulli's classical theory of probability.) However, in many cases, the probabilities of a class of events $A$ are needed to be estimated simultaneously based on the frequencies of their occurrence in a given sample, such that the convergence (in probability) of frequencies to probabilities will be *uniform* over all the events in the class $A$. I.e., that, for all $\epsilon, \delta > 0$, there exist $m_0$ such that for every $m > m_0$,

$$\Pr\left[\sup_{a \in A} |\nu^m(a) - p(a)| > \epsilon\right] < \delta. \tag{1.9}$$

If the set $A$ contains only finitely many events, then this requirement holds trivially, since we can apply Hoeffding bound to each of the events in $A$ separately in order to assure that

$$\Pr\left[|\nu^m(a) - p(a)| > \epsilon\right] < \delta/|A|, \ \forall a \in A, \tag{1.10}$$

[5] For many classes of hypotheses, the cardinality of the hypothesis space is infinite. In theses cases, the relevant measure of the entropy is the *Vapnik entropy*, which will be described in section 1.4.1.

and using the "union bound" we have:

$$\Pr\left[\sup_{a\in A}|\nu^m(a) - p(a)| > \epsilon\right] < |A|\delta/|A| = \delta. \tag{1.11}$$

as required. However, if the cardinality of $A$ is infinite, uniform convergence does not necessarily hold. Consider, for example, the class of sets:

$$a(\omega) = \{x \in [0, 2\pi] | \sin(\omega x) \geq 0\},\ \omega > 1.$$

Given a sample of $m$ points $x_1, \ldots x_m$, drawn according to a uniform distribution from the interval $[0, 2\pi]$, there is an infinite number of sets for which all of the $x$'s satisfy $\sin(\omega x) \geq 0$, which can be found by fitting $\omega$ to the sample such that:

$$\sin(\omega' x_i) \geq 0,\ i = 1 \ldots m.$$

However, if another sample is taken, then, for $\omega' >> 1$, only about half of the new $x'$s would satisfy $\sin(\omega' x) \geq 0$, which implies that the actual probabilities of these events should be close to $\frac{1}{2}$.

Conditions for uniform convergence of relative frequencies of events to their probabilities were studied by Vapnik and Chervonenkis[178]. Necessary and sufficient conditions for uniform convergence are stated in term of a function known as the Vapnik-Chervonenkis entropy and a number known as the Vapnik-Chervonenkis (VC) dimension.

### 1.4.1 The Vapnik-Chervonenkis entropy and the VC dimension

Given a sample $X^m = (x_1, x_2, \ldots x_m)$, each event $a \in A$ divides the instances in $X^m$ into two classes: the $x'$s for which $a$ occurs, and the instances for which it does not occurs. Denote $a(x) = 1$ if $a$ occurs for $x$ and $a(x) = 0$ otherwise, thus each event induces a binary string of length $m$. The maximal possible number of strings is smaller or equal to $2^m$. If $|A| < \infty$, then clearly the maximal possible number of different strings is also smaller than $|A|$ . In general, the class of events induces the set of strings: :

$$\Pi_A\left(X^{(m)}\right) = \left\{(a(x_1), a(x_2), ...a(x_m)) \in \{0,1\}^m |\ a \in A\right\}. \tag{1.12}$$

**Definition 1** *The quantity $H_{_A}\left(X^{(m)}\right) = \log\left(\left|\Pi_A\left(X^{(m)}\right)\right|\right)$ is called the entropy of the class of events $A$ on the sample $X^{(m)}$.*

**Definition 2** *Let the instances $x_1, x_2, \ldots x_m$ be drawn i.i.d. from a space $\mathcal{X}$ according to a distribution $\mathcal{D}$. The expectation value of the entropy of the class $A$ w.r.t. all the possible samples of length $m$,*
*$H^{\mathcal{D}}_{_A}(m) = E_{\mathcal{D}}\log\left(\Pi_A\left(X^{(m)}\right)\right)$ is called the Vapnik-Chervonenkis entropy of the class of events $A$ w.r.t the distribution $\mathcal{D}$.*

**Theorem 1 (Vapnik-Chervonenkis)** *A necessary and sufficient condition for uniform convergence of frequency of events to their probabilities when the samples are drawn according to a probability distribution $\mathcal{D}$ is that:*

$$\lim_{m \to \infty} \frac{H_A^{\mathcal{D}}(m)}{m} = 0. \tag{1.13}$$

However, this condition for uniform convergence depends on the particular distribution $\mathcal{D}$, which may not be known. Vapnik and Chervonenkis introduce another condition for uniform convergence, which is *distribution free*, and it is stated in terms of a combinatorial parameter known as the Vapnik-Chervonenkis (VC) *dimension*.

### The VC dimension

Let

$$\Pi_A(m) = \max_{X^{(m)}} \left| \Pi_A(X^{(m)}) \right|. \tag{1.14}$$

i.e., $\Pi_A(m)$ measures the *maximal* number of possible different subdivisions of any sample of size $m$. We say that the sample $X^{(m)}$ is *shattered* by $A$ if $|\Pi_A| = 2^m$.

The VC-dimension of $A$ is the maximal size of a sample which can be shattered. I.e.,

$$VCd(A) = \max_{X^{(m)}} \left\{ \Pi_A(X^{(m)}) = 2^m \right\}. \tag{1.15}$$

and if no such finite $m$ exists – then the VC-dimension of $A$ is infinite.

Since the VC-entropy is, by definition, always smaller than the VC-dimension, a finite VC-dimension assures uniform convergence independently of the underlying distribution $\mathcal{D}$ over the input instances. This result lies in the heart of the distribution-free PAC learning theory.

## 1.4.2 The metric dimension

Follows the work of Vapnik and Chervonenkis [178], Pollard[137] and Dudley[45], Haussler[70] introduced a condition for uniform convergence in terms of the *metric dimension* of a class. Given a class $A$, the metric dimension of $A$ is defined as:

$$\dim(A) = \lim_{\epsilon \to 0} \frac{\log \mathcal{N}(\epsilon, A)}{\log(1/\epsilon)} \tag{1.16}$$

A finite metric dimension is a sufficient condition for uniform convergence.

# Chapter 2

# Learning concepts and functions

## 2.1 The basic formal model of concepts learning

*"Do not become a slave of your model;"*

*Vincent Van Gogh, a letter to Theo van Gogh*

**Assumptions and Notations**

- The input instances $x \in \mathcal{X}$ are drawn randomly according to a fixed distribution $\mathcal{D}$.
- The concept space $\mathcal{C}$ and the hypothesis space $\mathcal{H}$ are spaces of Boolean functions over $\mathcal{X}$, $f : \mathcal{X} \to \{-1, 1\}$.
- There exists a *target concept* $c \in C$ which the learner attempts to learn.
  If $c$ is also restricted to be in $\mathcal{H}$, then $\mathcal{H}$ and $\mathcal{C}$ are equal. Throughout the following sections it will be assumed that this is the case, and unless otherwise stated, we will use $\mathcal{H}$ and $\mathcal{C}$ as synonyms.
- The learner has access to a training sample of $m$ input instances together with their labels:
  $$X^{(m)} = \{(x_1, c(x_1)), \ldots ((x_m, c(x_m))\},$$
  where the input instances are drawn according to a fixed but unknown distribution $\mathcal{D}$.
- The *training error* of a hypothesis $h$ w.r.t. a sample of labeled input instances, $X^{(m)} = ((x_1, c(x_1)), \ldots, (x_m, c(x_m))$ is defined as:
  $$\epsilon_t(h) = \frac{1}{2m} \sum_{i=1}^{m} |(x_i) - c(x_i)|. \quad (2.1)$$

- A hypothesis whose training error w.r.t. a given sample $X^{(m)}$ is 0, is said to be *consistent* with the sample. The set of all hypotheses in $\mathcal{H}$ which are consistent with a given sample $X^{(m)}$ is called *the version space* of $X^{(m)}$ and is denoted by $\mathcal{V}\left(X^{(m)}\right)$.

- The *generalization error* of a hypothesis $h$ is defined as the probability of disagreement between the hypothesis $h$ and the target concept $c$ on the label of an input instance $x$, drawn from $\mathcal{X}$ according to the probability distribution $\mathcal{D}$:

$$\epsilon_g(h) = \mathbf{Pr}_{x \in \mathcal{D}}\left[h(x) \neq c(x)\right]. \tag{2.2}$$

Given any $\epsilon, \delta > 0$ and a sample of $m(\epsilon, \delta)$ labeled input instances, $(x_1, c(x_1)), \ldots (x_{_M}, c(x_{_M}))$, the learning task is to find a Boolean hypothesis $h : \mathcal{X} \to \{-1, 1\}$, such that, with probability at least $1 - \delta$ over the samples, $\epsilon_g(h) < \epsilon$. If the existence of the above condition can be assured for *any distribution* $\mathcal{D}$ over the input instances, we say that the concept space $\mathcal{C}$ is PAC-learnable using $\mathcal{H}$. If the condition holds only when the input instances are drawn according to a specific distribution $\mathcal{D}$, we say that *$\mathcal{C}$ is $\epsilon$-$\delta$ learnable using $\mathcal{H}$ w.r.t. the distribution $\mathcal{D}$.*

## 2.2 Metric structures over the hypotheses space

A natural pseudo-metric structure over $\mathcal{H}$ can be defined as follows:

$$d_{_\mathcal{H}}(h, h') = \mathbf{Pr}_{\mathcal{D}}(x \in \mathcal{X} | h(x) \neq h'(x)). \tag{2.3}$$

The distance between two hypotheses $h, h' \in \mathcal{H}$ is defined as the probability that the two hypotheses would assign different labels to an input instance $x$, drawn from $\mathcal{X}$ according to $\mathcal{D}$. This is also the generalization error of $h$ when $h'$ is the target concept or the generalization error of $h'$ when $h$ is the target concept.

It is easy to see that this is indeed a pseudo-metric, since:

- $d(h, h) = 0 \quad \forall h \in \mathcal{H}$,

- $d(h, h') \geq 0, \quad \forall h, h' \in \mathcal{H}$, but in general it is possible that two different hypotheses would assign the same labels for all $x \in \mathcal{X}$, which implies that $d$ is a *pseudo-metric.*

- $d(h, h') = d(h', h) \quad \forall h, h' \in \mathcal{H}$.

- The triangle inequality: $d(h, h') \leq d(h, h'') + d(h'', h')$ $\forall h, h', h'' \in \mathcal{H}$ holds since if $h$ and $h'$ do not agree on a certain input instance, then either $h$ and $h''$ would not agree on this instance or else $h'$ and $h''$ would not agree on this instance.

The learning problem can now be re-defined in terms of the above pseudo-metric: Given any $\epsilon, \delta > 0$ and a sample of $M(\epsilon, \delta)$ labeled input instances, $(x_1, c(x_1)), \ldots (x_{_M}, c(x_{_M}))$, the learning task is to find a Boolean hypothesis $h : \mathcal{X} \to \{-1, 1\}$, such that:

$$\mathbf{Pr}\left[d(h, c) > \epsilon\right] < \delta. \tag{2.4}$$

## 2.3 The relation between learning and uniform convergence

If all the labels of the input instances in the sample are believed to be correct, then the best learning strategy is clearly to find a hypothesis that minimizes the training error $\epsilon_t$. However, the ultimate goal of the algorithm is to minimize the *generalization error* $\epsilon_g$. The basic condition for learnability is therefore that the training error *uniformly approximate* the generalization error, i.e., that for any $\epsilon, \delta > 0$ there exist $m_0(\epsilon, \delta)$ s.t. for any $m > m_0$ and any target concept $h_t$ in $\mathcal{H}$,

$$\mathbf{Pr}\left[|\epsilon_t(m) - \epsilon_g| > \epsilon\right] < \delta. \tag{2.5}$$

In particular, if it can be assured that the algorithm will find a hypothesis whose training error is zero, the generalization error would uniformly converge to zero, which is the basic requirement for learnability.

In the following sections several sufficient conditions for learnability will be described, either under specific distributions or under any distribution over the input instances.

### 2.3.1 Learning in a finite hypothesis space

The simplest case in which learnability is assured is the case where $\mathcal{H} = \mathcal{C}$ and there is a finite number of hypotheses in $\mathcal{H}$:

**Proposition 2** *If $|\mathcal{H}| < \infty$ then $\mathcal{H}$ is $\epsilon, \delta$ PAC learnable using a labeled sample of*

$$m \; > \; \frac{1}{\epsilon} \log \frac{|\mathcal{H}|}{\delta}$$

*inputs.*

**Proof:** Since the target concept $h_t$ is contained in $\mathcal{H}$, there is always at least one hypothesis in $\mathcal{H}$ whose training error is zero. We need therefore to show that $\epsilon_g$ uniformly converges to zero.

Since the input instances in the sample are i.i.d. random variables, the probability that a hypothesis $h$ whose distance from $h_t$ is greater than $\epsilon$ (an "$\epsilon$-bad hypothesis") would be consistent with a sample of size $m$ is, at most, $(1-\epsilon)^m$ :

$$Pr_{X^{(m)}}\left(h \in \mathcal{V}(X^{(m)}),\ error(h, h_t) > \epsilon\right) < (1-\epsilon)^m. \tag{2.6}$$

(In other words: the probability to find a sample $X^{(m)}$ that would be consistent with an "$\epsilon$-bad" hypothesis is smaller then $(1-\epsilon)^m$)

The probability that there exists an "$\epsilon$-bad hypothesis" in $\mathcal{H}$ that is consistent with a sample of size $m$ can therefore be bounded, using the "union bound," as follows:

$$Pr_{X^{(m)}}\left(\exists h \in \mathcal{V}(X^{(m)}),\ error(h, c) > \epsilon\right) < |\mathcal{H}|(1-\epsilon)^m. \tag{2.7}$$

An $\epsilon, \delta$ PAC learning is assured, providing that the total probability for the existence of an "$\epsilon$-bad consistent hypothesis" is smaller than $\delta$, which is assured provided that the sample size is such that:

$$(1-\epsilon)^m < \frac{\delta}{|\mathcal{H}|} \Longrightarrow m\log(1-\epsilon) < \log\frac{\delta}{|\mathcal{H}|} \Longrightarrow m \ > \ \frac{1}{\epsilon}\log\frac{|\mathcal{H}|}{\delta}, \tag{2.8}$$

where the inequality $\log(1-\epsilon) < \epsilon$ was used to derive the last inequality. Therefore, any finite hypotheses class is PAC learnable.

### 2.3.2 Learnability in an infinite hypotheses space

If $\mathcal{H}$ is infinite, there might be an infinite number of $\epsilon$-bad hypotheses, and in general one can not assure that for any $\epsilon$, $\delta > 0$, the probability of existence of an $\epsilon$-bad hypothesis that is consistent with a sample of size $m < \infty$ is smaller than $\delta$. However, under certain conditions learnability in the infinite $\mathcal{H}$ case can still be assured.

The problem is reminiscent of the problem of assuring a uniform convergence of frequencies to probabilities over an infinite class of events, based on a finite sample. It turns out that the conditions for uniform convergence are also necessary and sufficient to assure learnability[177][16], i.e., if the following condition on the VC-entropy holds:

$$\lim_{m\to\infty}\frac{H_{\mathcal{H},\mathcal{D}}(m)}{m} = 0. \tag{2.9}$$

Alternatively, if the VC dimension of $\mathcal{H}$, namely,

$$VCd(\mathcal{H}) = \max_{X^{(\ell)}}\left\{\Pi_{\mathcal{H}}(X^{(\ell)}) = 2^{\ell}\right\}. \tag{2.10}$$

is finite, than, given large enough sample of labeled input instances, a hypothesis which is consistent with the sample would have, with probability greater than $1-\delta$, an error which is smaller than $\epsilon$. (Note that the condition of a finite VC-dimension implies the condition in equation 2.9 for any distribution $\mathcal{D}$.)

A finite VC-dimension is a necessary and sufficient condition in order to assure a distribution-free PAC learning. If VCd($\mathcal{H}$) is finite, then the number of labeled input instances that are required to assure an error probability smaller than $\epsilon$ with a confidence level higher than $1-\delta$, (the *sample complexity*) is given by:

$$m \geq \left(\frac{1}{\epsilon}\log\frac{1}{\delta} + \frac{VCd(\mathcal{H})}{\epsilon}\log\frac{1}{\epsilon}\right). \tag{2.11}$$

### 2.3.3 Average-case error and sample complexity

The bound in equation 2.11 is a worst-case bound, both with respect to the possible distributions over the input instances and with respect to the possible targets in $\mathcal{H}$. Worst case bounds tend to be too pessimistic, and in many practical applications the actual sample complexity might be much smaller. In particular, if there is a prior $\mathcal{P}$ over the hypotheses in $\mathcal{H}$ s.t. the target hypothesis $h_t$ is drawn from $\mathcal{H}$ according to $\mathcal{P}$, then it can be shown that, given a sample of $m$ labeled input instances $X^{(m)}$, the expectation value of the error of the Bayes-optimal algorithm is bounded by[71]:

$$E_{h\in\mathcal{P},x\in\mathcal{D}}\left[\epsilon_{bayes}\right] \leq \frac{1}{2m} E_{h\in\mathcal{P},x\in\mathcal{D}}\left[H^{\mathcal{P}}\left(\Pi_m^{\mathcal{H}}(X^{(m)})\right)\right] \leq \frac{1}{2m} E_{x\in\mathcal{D}}\left[\log|\Pi_m^{\mathcal{H}}(X^{(m)})|\right]. \tag{2.12}$$

The average-case sample complexity is therefore bounded by

$$\frac{1}{2m} \cdot \text{VC-entropy}$$

.

## 2.4 The Dual learning problem

An input instance $x \in \mathcal{X}$ divides the hypotheses space $\mathcal{H}$ to two parts:

$$\mathcal{H}^+(x) = \{h \in \mathcal{H} | h(x) = +1\}, \quad \mathcal{H}^-(x) = \{h \in \mathcal{H} | h(x) = -1\}, \tag{2.13}$$

Similarly, a hypothesis $h \in \mathcal{H}$ divides the input space $\mathcal{X}$ to two parts:

$$\mathcal{X}^+(h) = \{x \in \mathcal{X} | h(x) = +1\} \quad \mathcal{X}^-(h) = \{x \in \mathcal{X} | h(x) = -1\} \tag{2.14}$$

This implies that there is a *duality* relation between the input and hypotheses spaces. If there exists a prior probability distribution $\mathcal{P}$ over the hypotheses space, then a pseudo-metric structure over the input space can be defined, similar to the pseudo-metric for the hypotheses space defined in equation 2.3:

$$d(x, x')_{\mathcal{X}} = \mathbf{Pr}_{\mathcal{P}}(h \in \mathcal{H} | h(x) \neq h(x')). \tag{2.15}$$

i.e., the distance between two input instances, $x, x' \in \mathcal{X}$ is the probability that a hypothesis $h$, drawn from $\mathcal{H}$ according to $\mathcal{P}$, would not assign the same label to $x$ and to $x'$. This distance reflects a measure of dissimilarity between input instances which is relevant to the learning task. Using this distance, a dual learning problem, of "learning" a specific input instances using its labels by a random sample of hypotheses, drawn from $\mathcal{H}$ according to $\mathcal{P}$, can be defined[138]:

**Input learning**
*Given an input space $\mathcal{X}$, a target input $x_0 \in \mathcal{X}$, a hypothesis space $\mathcal{H}$ with a prior $\mathcal{P}$ over the hypotheses and a sample $\mathcal{S}^m = (h_1(x_0), ..., h_m(x_0))$ of $m$ labeled hypotheses drawn from*

*$\mathcal{H}$ according to $\mathcal{P}$, find an input $x$ such that $d_{\mathcal{X}}(x_0, x) < \epsilon$ with probability at least $1-\delta$ for some $\epsilon, \delta > 0$.*

Within the context of this work, the dual learning problem serves as a main paradigm for the construction of faithful and balanced representations of input instances. However, dual learning problems have other usage: e.g., the dual DFA learning problem is important for learning description logic from sub-concepts and for learning function-free Prolog clauses from ground clauses[32], and the dual CNF learning problem, which will be discussed in chapter 6, is important for the understanding of learning from ambiguous information.

Two illustrative examples for dual learning problems are:

- Perceptron Learning: in the direct perceptron learning problem, the learning algorithm is given a sample:

  $$X^{(m)} = (\mathbf{x}_1, y_1, \ldots, \mathbf{x}_m, y_m)\,,\ \mathbf{x}_i \in R^n,\ y_i \in \{+1, -1\}.$$

  and attempts to find a vector $\mathbf{w} \in R^n$ s.t. $\text{sgn}(\mathbf{w} \cdot \mathbf{x}_i) = y_i$. In the dual learning problem, the learning algorithm is given a sample

  $$W^{(m)} = (\mathbf{w}_1, y_1, \ldots, \mathbf{w}_m, y_m)\,,\ \mathbf{w}_i \in R^n,\ y_i \in \{+1, -1\}.$$

  and attempts to find a vector $\mathbf{x} \in R^n$ s.t. $\text{sgn}(\mathbf{w_i} \cdot \mathbf{x}) = y_i$. The two problems are therefore completely equivalent.

- Axis aligned rectangles: in the direct problem there is an axis aligned target rectangle. The learning algorithm is given a sample:

  $$X^{(m)} = (\mathbf{x}_1, y_1, \ldots, \mathbf{x}_m, y_m)\,,\ \mathbf{x}_i \in R^2,\ y_i \in \{+1, -1\}$$

  where $y_i = 1$ if $\mathbf{x}_i$ resides inside the target rectangle and $y_i = -1$ otherwise, and attempts to find a rectangle that approximates the target rectangle. In the dual learning problem, there is a "target point" $\mathbf{x}_t$, and the learning algorithm is given a sample of $m$ random axis aligned rectangles, $r_1, \ldots r_m$, together with the set of labels:

  $$r_i(\mathbf{x}_t) = \begin{cases} 1 & \text{if } \mathbf{x}_t \text{ is inside } r_i \\ -1 & \text{otherwise} \end{cases} \tag{2.16}$$

  The learning algorithm is required to find a point $\mathbf{x}$ in $R^2$ which is close enough to $\mathbf{x}_t$ in the sense that there is only a small probability to find a rectangle $r_i$ such that $\mathbf{x}$ resides inside $r_i$ but $\mathbf{x}_t$ does not. (see fig. 2.1)

### 2.4.1 The VC dimension of the dual learning problem

The VC-dimension of the dual learning problems (sometimes called the *co-dimension*) obeys the inequalities[129]:

$$\begin{aligned} \log_2\left(dim_{VC}(\mathcal{X})\right) \le dim_{VC}(\mathcal{H}) \le 2^{dim_{VC}(\mathcal{X})}, \\ \log_2\left(dim_{VC}(\mathcal{H})\right) \le dim_{VC}(\mathcal{X}) \le 2^{dim_{VC}(\mathcal{H})}. \end{aligned} \tag{2.17}$$

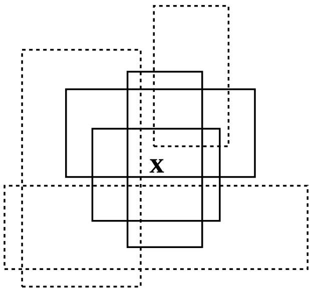


Figure 2.1: The dual problem of rectangle learning. Solid line: rectangles that contained the target $\mathbf{x}$ (positive examples), doted line - negative examples.

If the VC-dimension of one learning problem is finite, so is the VC-dimension of the other, which means that the direct and the dual learning problems are both PAC-learnable. Despite the possible exponential gap between the VC-dimensions of the two problems, for many natural learning problems, the two dimensions are similar: as we have seen, for perceptron learning the direct and the dual problem are equivalent, and therefore the corresponding VC-dimensions are equal. For partial parity functions the VC-dimension and the co-dimension are also the same. For geometric concepts in $R^2$, the co-dimension is the maximal number of hypotheses which constitute a *full Venn-diagram* (i.e., each hypothesis in a sample that can be shattered divides each region, generated by the intersections of previous hypotheses, into two parts (Fig 2.2)).

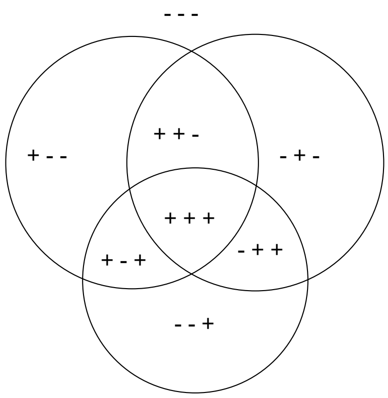


Figure 2.2: A full Venn-diagram

For axis-aligned rectangles the co-dimension is therefore 3, while the VC-dimension of the direct problem is 4. It is of great interest to characterize those problems more precisely. In this work we are primarily concerned with problems for which the two dimensions are at most polynomially apart.

## 2.5 Linearly separable problems

A Special class of learning problems consists of the *linearly separable problems*. If the input instances $\mathbf{x} \in \mathcal{X}$ can be considered as points in $R^n$, and the target concept is a Boolean function: $c : R^n \longrightarrow \{-1, 1\}$, then the problem is said to be linearly separable if there exists a vector $\mathbf{w} \in R^n$ and a real number $b$ such that:

$$c(x) = \operatorname{sgn}(\mathbf{w} \cdot \mathbf{x} + b), \quad \forall \mathbf{x} \in \mathcal{X}. \tag{2.18}$$

(the vector $\mathbf{w}$ is orthogonal to the separating hyperplane).

If $|\mathcal{X}| \leq n$ then the problem is clearly linearly separable. Cover (1965) [34] showed that, given a set of $m$ points in *general position* in $R^n$, (meaning that every subset of $n$ or fewer points are linearly independent) then the probability that the set is separable is:

$$\mathbf{Pr}(\text{Linearly separable}) = \begin{cases} 1 & \text{if } m \leq n+1 \\ 2^{1-m} \sum_{i=1}^{n} \binom{m-1}{i} & \text{otherwise} \end{cases} \tag{2.19}$$

If $(m-1)/2 < n$ then with probability greater than $1/2$ the set of $m$ points in linearly separable. If $m >> 2n$, the probability of linear separability is exponentially small. Therefore, the overwhelming majority of Boolean rules over a large input space $\mathcal{X}$ are not linearly separable. (See also Alon, N., P. Frankl, and V. Rödl, 1985.[5]).

Linearly separable problems were first studied by R. A. Fisher (1936) [48] in the context of statistical inference and were intensively studied ever since. During the late 1950's and early 1960's, linearly separable problems were studied as a framework for a neural-net brain model, where the coordinates of the vector $\mathbf{w}$ are considered as "synaptic weights". This work had its origin in McCulloch and Pitts famous paper from 1943[105], which was later developed by Rosenblatt[146] [149]. Rosenblatt introduced perceptron's learning algorithm, in which reinforcement rules are employed for changing the synaptic weights values and guaranteed error-free performance whenever it could be achieved.

Linearly separable problems are considered to be relatively simple to handle. In particular, if the problem is linearly separable, a separating hyperplane can be found using a polynomial linear-programming algorithm. Many other methods and algorithms for finding a separating hyperplane have been developed, many of these are described in Duda & Hart classical textbook[43].

As was mentioned earlier, most Boolean functions are not linearly separable. However, in many interesting real-life problem, it is possible to find a representation of the input instances that renders the problem linearly separable, and many useful algorithms are based on such transformations of representation. Multi Layer Perceptron (MLP), for one, attempts to transform the original representation of the input instances (the activation of the input layer) into a linearly separable one using the "hidden layer." In the following chapters several other transformations will be discussed.

## 2.6 Learning of real-valued functions

Concept learning can be considered a special case of learning a real-value functions, which consists of the following three components (after Vapnik[177]):

1. A generator (G) of random vectors $\mathbf{x} \in R^n$ drawn independently from a fixed, but unknown distribution function $F(x)$.
2. A supervisor, (S) who returns an output value $y$ to every input vector $\mathbf{x}$, according to a conditional distribution function $F(y|\mathbf{x})$, also fixed but unknown. (The case where the supervisor uses a function $y = f(\mathbf{x})$ can be considered a special case of this setting.)
3. A learning machine (LM) capable of implementing a set of functions $f(\mathbf{x}, \alpha)$, $\alpha \in \Lambda$, where $\Lambda$ is a set of parameters.[1]

The problem of learning is that of choosing from a given set of functions $f(\mathbf{x}, \alpha)$, $\alpha \in \Lambda$, the one that best approximates the supervisor's responses. The selection of the desired function is based on a training set of $m$ pairs $((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m))$, drawn according to $F(\mathbf{x}, y) = F(\mathbf{x})F(y|\mathbf{x})$.

In order to choose the best available approximation to the supervisor response, one measures the *loss* $L(y, f(\mathbf{x}, \alpha))$ between the response $y$ of the supervisor to a given input $\mathbf{x}$ and the response $f(\mathbf{x}, \alpha)$ provided by the learning machine. The *risk functional* is defined as:

$$R(\alpha) = \int L(y, f(\mathbf{x}, \alpha))dF(\mathbf{x}, y). \tag{2.20}$$

The goal of the learning machine is to find the function $f(\mathbf{x}, \alpha_0)$ that minimizes the risk functional $R(\alpha)$, using only the information in the sample $((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m))$.

Many learning problems can be addressed within this framework. Three of the most important problems are: regression estimation, concept learning, and density estimation.

### 2.6.1 Regression estimation

Let the supervisor responses be real values $y \in R$, and let $f(\mathbf{x}, \alpha)$, $\alpha \in \Lambda$ be a set of real functions that contains the regression function:

$$f(\mathbf{x}, \alpha_0) = \int y dF(y|\mathbf{x}). \tag{2.21}$$

The *regression function* is the function that minimize the risk functional 2.20 with the loss-function:

$$L(y, f(\mathbf{x}, \alpha)) = (y - f(\mathbf{x}, \alpha))^2. \tag{2.22}$$

[1]The elements $\alpha \in \Lambda$ are not necessarily vectors. They can be any abstract parameters. Therefore this setting is general enough to consider *any* set of functions.

in a situation where the probability measure $F(x,y)$ is unknown but the data $((\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_m,y_m))$ is given.

Conditions for learnability of real-valued functions require an introductions of several new concepts. Since this thesis does not deal with the learning of real valued functions, those conditions will not be addressed here. For a discussion of those conditions see, e.g. Vapnik (1995)[177] Alon, Ben-David, Cesa-Bianchi and Haussler (1993)[4]

### 2.6.2 Concept learning

The familiar problem of concept learning can be described within the framework of the above setting as follows: The supervisor's responses take only two values: $y \in \{0,1\}$. The set of functions $f(\mathbf{x},\alpha)$, $\alpha \in \Lambda$ are Boolean (or *indicator*) functions: $f(\mathbf{x},\alpha) \in \{0,1\}$, $\forall \mathbf{x}$, and the loss-function is:

$$L(y, f(\mathbf{x},\alpha)) = \begin{cases} 0 & \text{if } y = f(\mathbf{x},\alpha), \\ 1 & \text{if } y \neq f(\mathbf{x},\alpha) \end{cases} \tag{2.23}$$

For this loss function, the risk functional 2.20 determines the generalization error. The learning problem is to find a function that minimizes the generalization error when the probability measure $F(x,y)$ is unknown but the data $((\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_m,y_m))$ are given.

### 2.6.3 Density estimation

Density estimation (in the Fisher-Wald setting [49]) is an un-supervised learning problem, where the learning machine is given a set of vectors $(\mathbf{x}_1,\ldots\mathbf{x}_n)$ independently and identically distributed according to an unknown probability distribution $F(x)$, and is required to minimize the risk functional 2.20 with the following loss function:

$$L(f(\mathbf{x},\alpha)) = -\log f(\mathbf{x},\alpha). \tag{2.24}$$

# Chapter 3

# General learning algorithms

*I am the very model of a modern Major-General:*
*I've information vegetable, animal and mineral*

*Gilbert and Sullivan*

## 3.1 Introduction

The basic paradigm for concept learning which was introduced in section 2.1 implies that different learning problems in this setting basically differ only in their hypotheses spaces, the distribution over the input instances and the prior over the hypotheses (if one exists). This suggests that there may be a "meta-algorithm" that gets as an input an hypotheses space $\mathcal{H}$, a sample $X^{(m)}$ of $m$ labeled input instances and possibly also a prior $\mathcal{P}$ over the hypotheses space, and for any $\epsilon, \delta > 0$ outputs a hypothesis $h'$ (not necessarily in $\mathcal{H}$) such that the generalization error of $h'$ w.r.t. the target concept $h_t$ is smaller than $\epsilon$ with a confidence level higher than $1 - \delta$.

Based on known hardness results in computational complexity, cryptography, and computational learning theory, a meta-algorithm that can efficiently solve *any* learning problem does not exist. However, there do exist learning methods that perform well on a large class of learning problems. We refer to such algorithms "general learning algorithms."

Within the context of concept learning, the first attempt to construct a "general learning algorithms" is probably due to Rosenblatt (1962)[149], who suggested that each input instance $\mathbf{x}$ of a non-linearly separable problem will be transformed into vector $(\varphi_1(\mathbf{x}), \varphi_2(\mathbf{x}), \ldots \varphi_n(\mathbf{x}))$, where $\varphi_i$ is a randomly generated Boolean function of a small number of coordinates. Rosenblatt conjectured that if $n$ is large enough, then the set of transformed vectors will become linearly separable, and then the perceptron algorithm can be applied in order to find an "hypothesis" that correctly classifies all input instances.

At the time of Rosenblatt suggestion, the distinction between training error and generalization error was not very clear, and so it was not clear whether, and under what conditions, this scheme performs well. During the previous decade, training methods for multi-layer neural-networks, such as the celebrated "back-propagation"[148] algorithm, were developed. Multi-layer neural-networks can be considered as universal learning machines and are successfully applied in numerous domains. Related universal learning machines, in the spirit of Rosenblatt's original suggestion, are Radial-Basis Functions (RBF) networks [140], which will be discussed in section 3.5, and "the distributed method"[60], where the set of functions are "random perceptrons" (i.e., perceptrons whose weights are chosen at random) w.r.t. the inputs.

Within the context of computational learning theory, there are several successful paradigms of a "general learning algorithm." Some of these methods will be described in the following sections.

## 3.2 Support vector machines

### 3.2.1 Optimal separating hyperplane

Suppose that we are given a sample $X^{(m)} = \{(\mathbf{x_1}, c(\mathbf{x_1})), \ldots, (\mathbf{x_m}, c(\mathbf{x_m}))\}$, of $m$ labeled input instances, $\mathbf{x_i} \in R^n$, and we know that the problem is linearly separable, i.e., that there exists $\hat{\mathbf{w}} \in R^n$, $b \in R$ s.t. $c(\mathbf{x}) = \mathrm{sgn}\,(\hat{\mathbf{w}} \cdot \mathbf{x} + b)$, $\forall \mathbf{x} \in \mathcal{X}$. If no other restrictions are imposed on the separating hyperplane, then, in general, there exist infinitely many hyperplanes that would correctly classify the input instances in the sample. However, every hyperplane $\mathbf{w} \neq \hat{\mathbf{w}}$ would probably have a non-zero generalization error. Since the aim is to minimize the generalization error, an *optimal separating hyperplane* would be the hyperplane that, based on the information in the sample $X^{(m)}$, has the highest probability to have a minimal generalization error.

Vapnik and Chervonenkis [178] showed that an optimal separating hyperplane is the hyperplane that correctly classifies the input instances in the sample and that has the maximal *margin*, i.e., the hyperplane whose distance from the closest input instance is maximal. This notion of optimality seems intuitively very reasonable, since a large enough margin assures that small variations in the hyperplane coordinates would not change the classification of the sample.

It turns out that the optimal hyperplane is a linear combination of the vectors of the training set:

$$\mathbf{w_0} = \sum_{i=1}^{n} c(\mathbf{x_i}) w_i^0 \mathbf{x_i}. \tag{3.1}$$

Moreover, since the optimal hyperplane is determined by the inputs with the smallest margin, the only input vectors $\mathbf{x_i}$ that have nonzero coefficients $w_i^0$ in the above expansion are input vectors for which the margin is minimal, which are called *support vectors*.

The coordinates of optimal separating hyperplane can be found by solving a quadratic optimization problem:

Maximize:

$$\sum_{l=1}^{n} w_l - \frac{1}{2}\sum_{i,j=1}^{n} w_i w_j c(\mathbf{x}_i)c(\mathbf{x}_j)(\mathbf{x}_i \cdot \mathbf{x}_j). \tag{3.2}$$

under the constraints:

$$w_i \geq 0,\ i = 1, \ldots, n, \tag{3.3}$$

$$\sum_{i=1}^{n} w_i c(\mathbf{x}_i) = 0. \tag{3.4}$$

### 3.2.2 Support vector (SV) machines

The Support Vector (SV) machine, introduced by B. Boser, I. Guyon, and V. N. Vapnik, [18] [177] implements the following idea: it maps the input vectors $\mathbf{x}$ into a high-dimensional *features space* $Z$ through some nonlinear mapping chosen *a-priori*: $\mathbf{x} \in R^n \longrightarrow \mathbf{z}(\mathbf{x} \equiv (\varphi_1(\mathbf{x}), \ldots, \varphi_n(\mathbf{x})) \in R^d$ (where d is possibly infinite) In this space, a separating hyperplane is constructed. As was mentioned in section 3.1, the idea of mapping the input vectors into a high dimensional feature space where they could become linearly separable, was probably first introduced Rosenblatt during the late 1950's[149], and several learning algorithms use similar ideas in various disguises. The novelty of support vector machines lies in the manner in which they deal with two crucial problems:

1. How to find a separating hyperplane in the high-dimensional space that will generalize well?

2. How to treat computationally such a high-dimensional space?

The first problem is solved by constructing an *optimal hyperplane.* It turns out that in many cases the optimal hyperplane will generalize well, even if the dimensionality of the space is much higher than the number of training examples. In fact, it can be shown that, given a training sample of size $m$, the generalization error is bounded by:

$$\epsilon_g \leq \frac{E\left[\text{number of support vectors}\right]}{m-1} \tag{3.5}$$

where $E$ stands for expectation. Therefore, if the optimal hyperplane can be constructed from a small number of support vectors relative to the training set size, the generalization ability will be high – even in an infinite dimensional space. In many real life problems the number of support vectors is relatively small, and the generalization ability of SV machines is good.

The solution for the second problem is more complicated, and it is still under study. The key idea, which first enabled the practical usage of SV machines, was the observation made by Boser, Guyon and Vapnik (1992)[18] that, for constructing the optimal separating hyperplane in the feature space $Z$, one does not need to consider the feature space in *explicit form*. One only has to be able to calculate the inner product between support vectors and the vectors of the feature space:

$$(\mathbf{z}_i \cdot \mathbf{z}) \equiv K(\mathbf{x}_i, \mathbf{x})$$

where $K(\mathbf{x}, \mathbf{x}')$ is a symmetric positive definite kernel function that depends on the choice of the features and represents the scalar product in the feature space. This formulation made the computational problems of SV machines manageable. Support vector machines have by now found their use in an increasing number of applications, such as Optical Character Recognition (OCR)[177] and analysis of foreign exchange rate time series[128]. Support Vector Machines also introduce methods to deal with non-linearly separable cases and for learning in the presence of classification noise.

## 3.3 Weak learning and "boosting"

The notion of "weak learnability" was introduced by Kearns and Valiant[84]: let there be given a hypotheses space $\mathcal{H}$, a space of input instances $\mathcal{X}$, a distribution $\mathcal{D}$ over the space of input instances, and let $n$ be the size of the input instances. A learning algorithm $L$ is called a *weak PAC learning algorithm for $\mathcal{H}$* if there are fixed polynomials $p$ and $q$ such that $L$ outputs a hypothesis $h \in \mathcal{H}$ that, with probability at least $1/q(n, size(h))$, satisfies:

$$error(h) \leq 1/2 - 1/p(n, size(h)).$$

The requirement of a weak learning algorithm seems to be very modest: it should only be slightly better than a random guess. However, R. Schapire[154] was able to show that, if the weak learning algorithm can produce such a hypothesis for *any* distribution $\mathcal{D}$ over the input instances, then both the confidence level and the error rate can be "boosted" to an arbitrary accuracy $\epsilon$ with an arbitrary confidence level $1 - \delta$. This is done by invoking the weak learning algorithm repeatedly, each time with respect to a different distribution over the input instances, and using the resulting ensemble of "weak learners" in order to determine the label of a novel input instance.

The basic algorithm of Schapire was improved by Freund[52][53] , who presented a considerably more efficient "boost-by-majority" algorithm. Later, Freund and Schapire introduced an adaptive version of a boosting algorithm. This algorithm, known as "AdaBoost", is very nearly as efficient as "boost-by-majority." However, unlike "boost-by-majority," the accuracy of the final hypothesis produced by the **AdaBoost** algorithm depends on the accuracy of *all* the hypothesis returned by the weak-learning algorithm $L$, and so it is able to more fully exploit the power of the weak learning algorithm:

**Algorithm AdaBoost**

**Input:**

- Set of $N$ labeled examples $\{(x_1, c(x_1)), \ldots, (x_N, c(x_N)\}$.
- Distribution $\mathcal{D}$ over the examples.
- Weak learning algorithm $L$.
- An integer $T$ specifying the number of iterations.

**Initialize** the weight vector of the examples, $\mathbf{W}^1 : w_i^1 = \mathcal{D}(x_i),\ i = 1, \ldots, N$

**Do for** $t = 1, 2, \ldots, T$

1. set

$$\mathbf{P}^t = \frac{\mathbf{W}^t}{\sum_{i=1}^{N} w_i^t}$$

2. Call $L$, providing it with the distribution $\mathbf{P}^t$; get back a hypothesis $h_t$.
3. Calculate the error of $h_t$: $\epsilon_t = \sum_{i=1}^{N} p_i^t |h_t(x_i) - c(x_i)|$.
4. Set $\beta_t = \frac{\epsilon_t}{(1-\epsilon_t)}$.
5. Set the new weight vector to be:

$$w_i^{t+1} = w_i^t \beta_t^{1-|h_t(x_i)-c(x_i)|}$$

**Output** the hypothesis:

$$h_f(x) = \begin{cases} 1, & \text{if } \sum_{t=1}^{T} \left(\log \frac{1}{\beta_t}\right) h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \log \frac{1}{\beta_t} \\ 0 & \text{otherwise} \end{cases} \tag{3.6}$$

The algorithm adjusts adaptively to the errors of the weak hypotheses returned by $L$. It has a main loop which is iterated $T$ times. Each time it defines a distribution $\mathbf{P}^t$ over the samples, according to the current weight vector $\mathbf{W}^t$. It then feeds this distribution to $L$ and gets back a hypothesis $h_t$ whose error is $\epsilon_t \leq 1/2 - \gamma$ for all $t$. However, $\gamma$ need not be known before the construction of the algorithm. The parameter $\beta_t$ is chosen as a function of $\epsilon_t$ and is used for updating the weight vector. The update rule reduces the probability assigned to those examples on which the hypothesis make good prediction and increases the probability of the examples on which the prediction is poor. The final hypothesis generated by the algorithm is a weighted average of the $T$ hypotheses generated by $L$.

One way to view the boosting method is as a method for a representational shift, where each input instance is represented according to its labels by the $T$ "weak hypotheses." Using this representation, the problem becomes linearly separable, though on a possibly high-dimensional space. This is reminiscent of the "support vector machine" method. The relations between the methods of boosting and of support vector machines in terms of "maximal margin classifiers" are analyzed by Schapire, Freund, and Bartlett in [156].

## 3.4 Decision trees

Decision tree algorithms have been studied throughout machine learning and statistics as a nonparametric approach to data modeling and as a simple, yet extremely efficient method for classification and regression[12][142]. A *decision tree* is a binary tree where each internal node is labeled with a binary variable (which may be the result of a test), and each leaf is labeled with either 1 or 0. Each decision tree defines a boolean function. An assignment to the variables determines a unique path from the root to a leaf: at each internal node the left (right) edge to a child is taken if the value of the node variable in the assignment is 1 (0). The value of the tree function at the assignment is the value at the leaf reached. The *depth* of a decision tree is the length of the longest path from the root to the a leaf. If all the paths have the same length, we say that the decision tree is *balanced* (see figure 3.1). Decision trees found their use in many applications, from medical diagnosis to elementary

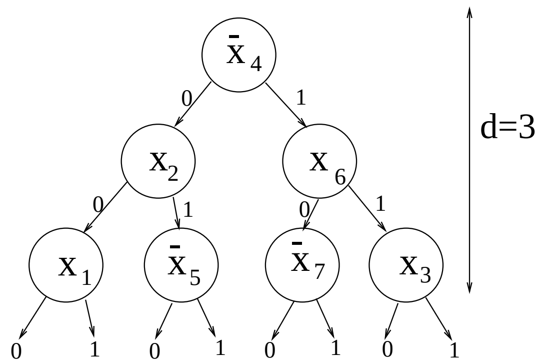


Figure 3.1: A balanced decision tree of depth 3. variables with bar are negated

particle detection[12].

Most of the machine learning algorithms that use decision trees, employ an heuristics for building top-down decision trees from labeled sample data. Such algorithms grow a tree from the root to the leaves by repeatedly replacing an existing leaf with an internal node, thous "splitting" the data reaching this leaf into two new leaves, and thereby reducing the empirical error on the given sample. Various heuristics differ mainly by the methods and the criteria for replacing leaves by nodes.

While in general it is hard to find a decision tree that give perfect classification, it is relatively easy to construct "weak learners" using decision trees. This make decision trees a perfect candidate for boosting techniques, and Breiman, one of the pioneers of decision-trees modeling, crowned AdaBoost with decision trees "the best of the shelf classifier in the world". (NIPS workshop, 1996)

## 3.5 Fourier transform methods

Given a set $\mathcal{F}$ of Boolean functions of $n$ variables, $f : \{0,1\}^n \rightarrow \{-1,1\}$, the inner product of two functions is defined as

$$\langle f_i, f_j \rangle = 2^{-n} \sum_{\mathbf{x} \in \{0,1\}^n} f_i(\mathbf{x}) f_j(\mathbf{x}) = E[f_i \cdot f_j],$$

where $E$ is the expectation with respect to the uniform distribution. The basis for the Fourier transform is defined as follows: to each $\mathbf{z} \in \{0,1\}^n$, there corresponds a *basis function*

$$\chi_z(\mathbf{x}_1, \ldots, \mathbf{x}_n) = (-1)^{\sum_{i=1}^{n} \mathbf{x}_i \mathbf{z}_i}.$$

(i.e., parities of subsets of inputs variables, which are determined by the non-zero elements of $z$). Any function of $n$ Boolean inputs can be uniquely expressed as a linear combination of the basis functions. For a function $f$, the $\mathbf{z}^{th}$ Fourier coefficient are computed by: $\hat{f}(\mathbf{z}) = \langle f, \chi_z \rangle$ and $z$ is called the *coefficient-name* of $\hat{f}(\mathbf{z})$. A *t-sparse* function is a function that has, at most $t$ non-zero Fourier coefficients.

The work of Linial, Mansour and Nisan[97] was the first to point out the connection Between the Fourier spectrum and learnability. They presented a quasi-polynomial time (i.e., $O\left(n^{poly-log(n)}\right)$) algorithm for approximate learning of polynomial size constant-depth circuits (where the approximation is with respect to the uniform distribution). Aiello and Mihail (1991)[3] introduced a polynomial time learning algorithm for learning both probabilistic decision lists and probabilistic read-once decision trees with respect to the uniform distribution. Kushilevitz and Mansour (1991)[92] use the Fourier representation to derive a polynomial time learning algorithm for decision trees with respect to the uniform distribution. The algorithm is based on a procedure that finds the significant Fourier coefficients. The algorithm was further improved by Mansour and Sahar (1995) [103], and was used for learning polynomial size DNF.

Fourier transform methods are universal in the sense that they do not make any assumptions on the function it is learning. One can apply it to *any* function and hope to obtain "large" Fourier coefficients. The prediction function simply computes the sum of the coefficients with the corresponding basis function and compares the sum to some threshold. The procedure is also immune to some noise and will be able to operate even if a fraction of the examples are maliciously misclassified. However, since the basis functions were selected independently of the learning problem, it does not employ any problem specific knowledge, which may reduce the efficiency of the learning algorithm.

## 3.6 Radial basis functions

Radial basis function (RBF) methods have their origin in techniques for performing exact interpolation of a set of data points in a multi-dimensional space [140]. In the context of neural network, the paradigm of RBF was first introduced by Broomhead and Lowe [20]. Poggio and Girosi[136] later showed that RBF can be explained within the context of regularization theory. RBF networks can serve both for function approximation and for classification. (In their use for classification, RBF can be considered as a bridge between the non-parametric k nearest-neighbors methods and parametric classification methods.) They have proven to be useful in a variety of applications such as time series analysis [150] [81] classification of seismic events [25], handwritten digit recognition [94], speech recognition[95] adaptive control [153], spectral estimation [118] and many other applications.

The main idea behind the radial basis function scheme is to map a set of $n$ input vectors $\{\mathbf{x} \in R^d\}$ into a set of outputs $\{\mathbf{y} \in R^{d'}\}$, which should approximate a desired set of outputs, $\hat{\mathbf{y}}_1, \ldots, \hat{\mathbf{y}}_n$, using a mapping function that is a linear superposition of a set of functions, $\varphi_1, \ldots, \varphi_m$. The arguments of these functions are the distances between the input vectors and a set of pre-defined centroids $\mathbf{c}_1, \ldots, \mathbf{c}_m$:

$$y_i(\mathbf{x}) = \sum_{j=1}^{n} \omega_{ij} \varphi_j \left( ||\mathbf{x} - \mathbf{c}_j|| \right). \tag{3.7}$$

The dimension of the outputs, $d'$, is determined by the specific task at hand: for interpolation, we usually have $d' = d$. For multi-class classification tasks, $d'$ is usually equal to the number of classes, and the classification of the input instance is determined by the most active $y$ (a *"winner-takes-all" method*). In the binary classification task, it usually suffices to take $d' = 1$, and to assign one label to $\mathbf{x}$ if $y(\mathbf{x}) > \Theta$ and the other label if $y(\mathbf{x}) < \Theta$ for some threshold value $\Theta$.

In the most basic RBF scheme, the centroids are just the input instances, $\mathbf{x}_1, \ldots, \mathbf{x}_n$ and the functions $\varphi_1, \ldots, \varphi_n$ are all the same. Other schemes uses different functions from the same parametric family.

## 3.7 Gaussian process classifiers

Gaussian processes are useful tools for regression estimation within the framework of regularization theory (see next section). Lately, methods for using Gaussian processes for classification were suggested (Gill and MacKay (1997)[68] Neal (1995)[122]). This framework assumes the existence of a class of functions over the input instances $\mathbf{x}$, $a(\mathbf{x}, \alpha)$, $\alpha \in \Lambda$ (where $\alpha$ is a set of hyper-parameters which characterize the function) and a function in the class $a(\mathbf{x}, \alpha')$ that model the log-likelihood ratio of the two possible classifications of $\mathbf{x}$, $c(\mathbf{x}) = 1$ and $c(\mathbf{x}) = -1$:

$$a(\mathbf{x}, \alpha') = \log \frac{P(c(\mathbf{x}) = 1)}{P(c(\mathbf{x}) = -1)}. \tag{3.8}$$

and thus:

$$P(c(\mathbf{x}) = 1 | a(\mathbf{x}, \alpha')) = \frac{1}{1 + exp(-a(\mathbf{x}))}. \tag{3.9}$$

For the learning process, it is assumed that there exists a prior over the set of parameters $\alpha$. The set of parameters might be the set of weights of a neural network or the set of coefficients in a linear expansion in terms of a set of a pre-defined functions: $a(\mathbf{x}, \alpha) = \sum_i w_i^\alpha \varphi_i(\mathbf{x})$, and then a prior probability distribution $P(\alpha)$, which is traditionally taken to be Gaussian, can be placed over the parameters[102]. Alternatively, the function $a(\mathbf{x})$ can be modeled directly using a Gaussian process[184]. This involve modeling the joint distribution of $a(\mathbf{x}_1), \ldots, a(\mathbf{x}_m)$ with a Gaussian distribution:

$$P(\mathbf{a}_m | \alpha) = \frac{1}{Z} \exp\left( -\frac{1}{2} \mathbf{a}_m^T \mathbf{C}_m^{-1} \mathbf{a}_m \right). \tag{3.10}$$

where $\mathbf{a}_m = (a(\mathbf{x}_1), a(\mathbf{x}_2), \ldots, a(\mathbf{x}_m))$, $Z$ is the normalization factor and $\mathbf{C}_m$ is an appropriate positive definite covariance matrix that is a function of the inputs $(\mathbf{x}_1, \ldots, \mathbf{x}_m)$ and the set of hyper-parameters $\alpha$.

Within this framework, several techniques for producing Gaussian process classifiers were suggested, such as variational methods[68] and Monte Carlo methods[122]. The authors claims that their results are comparable to the best of current classification methods.

## 3.8 Multiple task learning and inductive transfer

The research in the area of inductive transfer addresses the question of how learning can be enhanced by using knowledge from multiple learning tasks. Inductive transfer can reduce both the sample complexity and the computational complexity. Recently, the field of transfer in learning has received considerable interest within the machine learning community[141].

In general, the presence of more than one learning task provides a synergistic advantage and allows to gain domain specific knowledge such as invariance, which can be exploited when learning other tasks. Research on transfer in inductive learning seeks to find computational mechanisms for learning and transferring knowledge across different learning tasks. This is usually achieved by learning tasks in parallel while using a shared representation[22].

A Bayesian paradigm of “learning to learn” by sampling from multiple tasks was introduced by J. Baxter [11] and was applied to the problem of learning a common feature set or equivalently a low-dimensional representation of the “environment” of related tasks. Baxter’s method has several lines of similarity to the faithful representation scheme which will be discussed in chapter 4.

## 3.9 Summary

In this chapter, several existing “universal” scheme for learning were described. Universal learning scheme does not depend too much on the specific problem at hand, and therefore can be applied to a large class of learning problems. In particular, such methods can be useful in cases where problem-specific methods do not exist. However, in many cases knowledge of the specific details of the problem can greatly enhance the performance of the learning algorithm, and so universal algorithm that does not take problem specific knowledge into account (such as Fourier transform methods) is, in general, sub-optimal. In the next chapter we introduce a universal learning method that does incorporate knowledge about the specific problem at hand.

# Part II

# Faithful Representations in Learning

# Chapter 4

# Faithful representations in learning.

*"Learning is but an adjunct to ourself."*

*W. Shakespeare, Love Labour's Lost*

## 4.1 Introduction

Among the most crucial factors for successful learning are the way in which input instances are represented to the learning system and the way in which the hypotheses are represented inside the system. These two problems are addressed in this chapter.

### 4.1.1 Representation of input instances

Representation of input instances is a long standing research topic in machine learning. Eventually, digital computers can only process information in a digital form, and information regarding physical objects need to be transformed into numbers. Those numbers need to convey information regarding the actual properties of the input instances that might be relevant for the learning task. This process is known as *pre-processing* or *feature extraction.* In the end of this process each input instance can be encoded as an array $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, where each of the components $x_1, \ldots, x_n$ is a number describing an attribute or a feature of the input instances. E.g., in a medical diagnosis system, each component may describe the existence of a symptom ($x_i = 1$ if the symptom exists and $x_i = 0$ otherwise) or a quantitative result of a test (such as body temperature or blood pressure). Some of those properties may not be relevant for the learning task, and their presence may increase both the number of input instances required in order to achieve a good generalization performance (the *sample complexity*), and the number of computational steps required for the learning task (the *computational complexity*). It is therefore advantageous to *select* the features that are actually needed for the learning task. This process is called *feature selection.* In some learning tasks, features may have a clear semantic meaning, as might be the case in some medical diagnosis problems, where the features may be the symptoms of a certain disease. In other cases, features can be extracted and selected automatically, without an explicit reference to their meaning, by applying various processing methods to the raw input instances. For instance, for acoustic signals, it is sometimes advantageous to represent the signal as a collection of frequencies, and then to use methods such as principal component analysis in order to extract and select the prominent features.

Even if the input instances are represented in a manner that conveys all the relevant information, and nothing but relevant information, it still does not necessarily mean that the representation is indeed optimal: the same information can still be encoded in infinitely many ways, and the *computational efficiency* of the learning algorithm depends, in general, on the way in which the information is encoded. (As an extreme example, consider the case where the learning problem is to find a natural number $a$ using a set of $m$ labeled input instances: $((x_1, c(x_1)), \ldots, (x_m, c(x_m))$, where $c(x_i) = \text{sgn}(x_i - a)$. If the representations of the input instances $x$ are encrypted using a scheme such as a RSA encryption, then it can be shown[84] that, under the conventional cryptographic assumptions, there can not exist an efficient learning algorithm even if all the relevant information exists.)

The representation of the input instances can therefore effect both the sample complexity and the computational complexity of the learning algorithm. An ideal representation scheme will be one that minimizes both the sample complexity and the computational complexity of the learning algorithm, either with respect to a specific target concept or with respect to a family of possible target concepts that need to be learned using the same set of inputs.

The distinction between the construction of the representation and the learning algorithm itself is not sharp. Many successful learning methods simultaneously transform the representation (in a lower hierarchical stages) and use the new representation for the final classification. Feed-forward multi-layer neural networks, for one, can be considered as such a scheme, and the activation pattern in each but the last layer is regarded as a representation of the input instances for the next layer. Indeed, Rumelhart, Hinton, and Williams (1986)[148] named their celebrated paper, which popularized the "back-propagation" algorithm, "Learning representations by back-propagating errors." Boosting methods, which were described in section 3.3, can also be considered as an example of a transformation of the representation induced by the learning algorithm: in the final stage of the algorithm, the algorithm classifies each input instance according to the label it has been given by the "weak learners," and therefore the outputs of the weak learner for a given input instance can be considered as a representation of the input instance (similar to the last "hidden layer" in a multi-layer feed-forward neural network). The construction of this representation is, again, a crucial part in the mechanism of the learning algorithm.

Representational shifts can be divided into two main categories: *target-dependent* shifts, in which the representational shifts are constructed for, or induced by, a specific target concepts, and *target-independent* shifts, which should be good enough for a wide range of target concepts in a specific domain. The representational shifts induced by the back-propagation and the boosting algorithms are examples for target-dependent shifts. Representation schemes such as projection to the high-dimensional "feature space," which is used in support vector machines (section 3.2) or representation of inputs using their Fourier transform (cf. section 3.4) on the other hand, does not depend on the specific target concept, and need not be learned.

As a result, there is a considerable redundancy in the representation in which the learning algorithm is required to handle.

The size of the representation is of great importance for the learning process. In general, lower dimensional representations require less computational resources and has a lesser sample complexity, and, indeed, until recently, most of the efforts were devoted to the reduction of the size of the representation, using methods such as principal components analysis and multi-layer neural networks. However, using a high-dimensional representation of input instances tend to make decision surfaces simpler, and in many cases the problem become linearly separable. As discussed in chapter 3, both boosting methods and support vector machines relay on the use of a high-dimensional representation of the input instances, while maintaining a reasonable sample complexity due to the use of an "optimal separating hyperplane".

### 4.1.2 Representation of the hypotheses space

In the standard models for concept learning, hypotheses are considered as sets of input instances. As such, they may be represented in many possible ways. The manner in which the hypotheses are represented does not have a direct effect on the sample complexity, but it can have an immense effect on the computational complexity of the learning process: learning is, in a sense, a search in the hypotheses space, and the manner in which the hypotheses are represented can greatly effect the efficiency of the searching process.

Within the context of PAC learning, the importance of hypothesis representation was first explored by Pitt and Valiant [139] who showed that $k$-term DNF is not efficiently PAC learnable using a hypothesis class of $k$-term DNF (i.e., the computational complexity of finding a consistent $k$-term DNF hypothesis is exponential) but is efficiently PAC learnable using $k$-CNF. The issue of representation of concept classes by embedding them in other classes was raised already by Littlestone and Warmuth[98] and later on by Floyd and Warmuth[50] and by Pitt and Warmuth[138] who raised the issues of *exact* embedding and of sample compression by embedding of learning classes.

In the Bayesian learning model, different representations can alter the "natural" prior over hypotheses. For example, if the hypotheses are sections of the circumference of a given circle, the hypotheses can be represented in terms of the corresponding chords or in terms of the corresponding central angles. A uniform prior over the length of the chords in the range $[0, 2r]$ is, of course, quite different from a uniform prior over the width of the angle in the range $[0, 2\pi]$. Such a shift in the prior over the hypotheses space may change the average sample complexity[72] (though not by more then a constant factor.)

It is not clear how to define and construct good representations of input instances and hypotheses. In the following section we suggest a definition and a construction method for *faithful and balanced binary representations*, – properties that seems advantageous while considering representations. In chapter 5 several learning algorithms which use such representations will be discussed.

## 4.2 Faithful and balanced representations

### 4.2.1 The learning model

Given a hypotheses space $\mathcal{H}$ of binary hypotheses and an input space $\mathcal{X}$, faithful and balanced representations of input instances and hypotheses will be defined within the framework of the following (Bayesian) learning model:

- The input instances $x \in \mathcal{X}$ are drawn independently according to a fixed distribution $\mathcal{D}$.
- The hypothesis space $\mathcal{H}$ is a space of Boolean functions over $\mathcal{X}$, $h : \mathcal{X} \to \{-1, 1\}$.
- The target hypothesis $h_t \in \mathcal{H}$ is drawn from $\mathcal{H}$ according to the (prior) distribution $\mathcal{P}$
- The learner has access to a training sample of $n$ labeled input instances $\{(x_1, h_t(x_1)), \ldots ((x_n, h_t(x_n))\}$, independently drawn from $\mathcal{X}$ according to the fixed but unknown distribution $\mathcal{D}$.
- The learner can sample the hypotheses space $\mathcal{H}$ according to the distribution $\mathcal{P}$.

Using the standard PAC definitions, the learning task is stated as follows; given any $\epsilon, \delta > 0$ and a sample of $m(\epsilon, \delta)$ labeled input instances, $(x_1, h_t(x_1)) \ldots (x_n, h_t(x_n)))$, find a Boolean hypothesis $h : \mathcal{X} \to \{-1, 1\}$, such that with probability at least $1 - \delta$ over the samples,

$$\mathbf{Pr}_x \left[ h(x) \neq h_t(x) \right] < \epsilon. \tag{4.1}$$

### 4.2.2 Faithful and balanced representations: from "objective" to "subjective" representations

Given an instance sample $\mathcal{S}_n = \{x_1, \ldots, x_n\}$, a *vector representation* of a hypothesis $h$ w.r.t. $\mathcal{S}_n$ is defined as:

$$r_{\mathcal{S}_n}(h) = (h(x_1), \ldots, h(x_n)).$$

$r_{\mathcal{S}_n}(h)$ is a vector of $\pm 1$. The normalized Hamming distance between the representations of two hypotheses is the *training error* of one hypothesis w.r.t. the other on the sample $\mathcal{S}_n$. The average of this training error over all the possible samples is, by definition, the *generalization error*. Assume that VC-dimension of $\mathcal{H}$ is $D_h < \infty$, then, using the uniform convergence property of frequencies to probabilities in finite VC-dimension classes[178], if:

$$n = O\left( \frac{D_h}{\epsilon^2} \left( \log \frac{1}{\epsilon} + \log \frac{1}{\delta} \right) \right)$$

then the normalized Hamming distance between the representations of hypotheses $h_i$ and $h_j$, $\frac{1}{n}ham(h_i, h_j)$, is uniformly an $\epsilon, \delta$ approximation for the generalization error of one hypothesis w.r.t. the other. Namely,

$$\mathbf{Pr}\left[\left|\frac{1}{n}ham(h_i, h_j) - \epsilon_g(h_i, h_j)\right| > \epsilon\right] < \delta$$

The distance between the representations therefore *approximately preserves the generalization error topology of the hypotheses class.* We call such mappings *an $\epsilon, \delta$ faithful representations.* The scalar product of the representation vectors, is directly related to the Hamming distance, as

$$\frac{1}{n}\mathbf{x_1} \cdot \mathbf{x_2} = 1 - \frac{2}{n} \cdot ham(x_1, x_2).$$

Thus the Euclidean topology also approximates the generalization error topology.

Similar construction can be applied to the instance space by considering *the dual learning problem* described in section 2.4, where a target *input instance $x_t$* is learned by its labeling of randomly drawn hypotheses. Denote by $D_x$ the VC-dimension of the dual problem. Then, given $\epsilon, \delta > 0$ and $m > \frac{D_x}{\epsilon^2}\left(\log\frac{1}{\epsilon} + \log\frac{1}{\delta}\right)$, a random sample ${\mathcal{S}_\mathcal{H}}^m$ of $m$ hypotheses is drawn from $\mathcal{H}$ according to $\mathcal{P}$. The vector representation of the input instance $x$ w.r.t. $\mathcal{S}_m$ is defined as

$$r_{\mathcal{S}_m}(x) = (h_1(x), \ldots, h_m(x))) \ .$$

Clearly, such representations are also $\epsilon, \delta$ *faithful* with respect to the generalization error of the dual problem.

If we can faithfully embed the instance space $\mathcal{X}$ in ,$\mathcal{X}'$ and the hypothesis space $\mathcal{H}$ in $\mathcal{H}'$ – the following diagram becomes approximately commutative, up to a *global isometry* of either $\mathcal{X}'$ or $\mathcal{H}'$.

$$\begin{array}{ccc}
\mathcal{X} & \text{—}\ L_\mathcal{H} \longrightarrow & \mathcal{H} \\
| & & | \\
r_x & & r_h \\
\downarrow & & \downarrow \\
\mathcal{X}' & \text{—}\ L_{\mathcal{H}'} \longrightarrow & \mathcal{H}'
\end{array}$$

There are two routes from $\mathcal{X}$ to $\mathcal{H}'$ in this diagram. A sample $x^{(m)} \subset \mathcal{X}$ can be used to learn a target $h_t \in \mathcal{H}$ via a learning algorithm $L_\mathcal{H}$ that generates an hypothesis $h(x^{(m)}) \in \mathcal{H}$. The quality of the algorithm is determined by the distance $d = d(h_t, h(x^{(m)}))$. Alternately, we can first embed the sample in $\mathcal{X}'$, $r_x(x^{(m)}) \subset \mathcal{X}'$ and then apply the learning algorithm $L_{\mathcal{H}'}$ to generate an hypothesis $h'(r_x(x^{(m)})) \in \mathcal{H}'$. The quality of this hypothesis as an approximation to the target $h_t$ is measured by the distance $d' = d(r_h(h_t), h'(r_x(x^{(m)})))$. The alternative route, through the embedding, is useful only if $d' \approx d$. In this case we may avoid possible computational hardness of the *proper*[1] algorithm $L_\mathcal{H}$ by a the corresponding *improper* $L_{\mathcal{H}'}$, provided that we can find *efficient* faithful embedding $r_x$ and $r_h$.

[1]Given a learning problem and a hypotheses space, a proper learning algorithm is allowed to use only hypotheses from the given hypotheses space, while improper learning algorithm is not restricted to use any specific hypotheses space

### 4.2.3 Some observations regarding the faithful representations

- The above representation scheme essentially translate the original "objective" representation of the input instances to a representation in terms of a class of hypotheses that is chosen by the learner – a "subjective" representation. As such, this representation may be more suitable for the learning task at hand. Similarly, the hypotheses are represented in term of the inputs – a "data driven" representation.

- By construction, this topology preserving *Euclidean embedding* of the instances have the property that inputs with "close" representations get, with high probability, the same label by a random target hypothesis. Since inputs have different labels only if they reside in opposite sides of the decision surface, we may expect that, using this embedding, the surface area of the decision surfaces should be smaller compared with the surface area of the decision surfaces using a different embedding, in which there is no direct relation between the distance between inputs and the probability that their labels be equal (Fig .4.1). Since smaller surface area usually corresponds to a simpler decision surface, we can expect that the use of faithful representations would (in general) tend to make the decision surface simpler.

- The random representations also have the nice property of being "balanced" in the sense that all the coordinates of the vectors are statistically equivalent. This scheme is therefore also a *"pre-whitening"* of the coordinates.

- The faithful representations are also "hologramic" in the sense that large enough random subset of coordinates contained, approximately, the same amount of information as other random subsets of the same size. Such representations therefore posses an inherent robustness to "noise" in the representation of the inputs. In addition, if there is a noise in the coordinates of the input instances themself, such that $\mathbf{x} \to \mathbf{x} + \delta\mathbf{x}$, the representation will change only at the hypotheses were the noise cause the input instances to cross their decision boundary. In general, this would promote the resistance of representation to noise in the inputs.

- The representation scheme also provides an interesting insight into distributed representation schemes: a distributed representation of input instances is in the core of connectionistic learning and is, most probably, a prevalent scheme in biological neuronal systems[47]. Determining the size of the representation in terms of the VC-dimension of the *dual learning problem* allows to define the required size of a representation that would preserve, with high probability, most of the relevant information.

### 4.2.4 Related representation schemes

Representing the input instances as vectors, using their values according to a pre-defined set of functions, is a well known procedure in approximation and pattern-recognition theories, and it has been used for quite a long time[44]. As was mentioned in section 3.1, the first

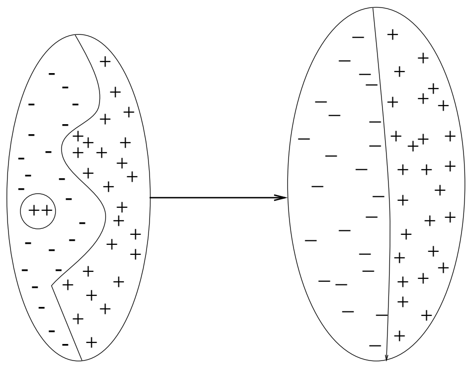

Figure 4.1: Schematic transformation of the bounderies due to the shift in the representation

attempt of using such a representation for classification is probably due to Rosenblatt[146]. During the previous decade, representations that use sets of Radial-Basis Functions (RBF) for approximation and classification were introduced and proved to be very useful [140]. In recent years there has been an increasing interest in such representations from several, seemingly independent directions:

- Support vectors machines[177], discussed in section 3.2, where inputs are mapped into a high dimensional feature space through some nonlinear mapping chosen *a-priori*:

  $$\mathbf{x} \in R^n \longrightarrow \mathbf{z}(\mathbf{x} \equiv (\varphi_1(\mathbf{x}), \ldots, \varphi_n(\mathbf{x})) \in R^d$$

  (where d is possibly infinite). In this space, an optimal separating hyperplane is constructed. As explained in section 3.2, the size of such representations is very large (possibly infinite), but the problem remains manageable due to the fact that effectively, only a limited number of "support vectors" need to be considered.

- From the standpoint of regularization theory, the methods of *Gaussian processes classifiers* (section 3.7) and *sparse approximation* [26] also use a vector representation with a set of basis functions taken from a large *dictionary*. These methods often appear in conjunction with the use of over-complete or redundant representations and the candidate approximations are linear superpositions of a small number of base functions. It can be shown that such methods are, in many ways, essentially equivalent to the Support Vector Machine method [62].

- In the Fourier transform method, discussed in section 3.5, the set of functions that are used for the representation are "partial parity" functions, independently of the learning problem.

- A distributed representation method, where the set of functions that are used for the representations are perceptrons with random weights, was introduced by Gallant and Smith, (1987) [60],[59].

- Representation of the inputs, using a random sample of classifiers that are used in related tasks ("sampling an environment of classifiers") was suggested by Baxter[10]. Baxter's method, which does not explicitly concern the dual learning problem, resemble our method in several aspects, but is based mainly on using many related learning tasks and does not fully explore the problem of linear separability, nor other learning algorithms using such representations that are studied in this work.

- Intrator and Edelman[76] have investigated a method for learning a low-dimensional representations via the use of multi-class labels. The motivation for their work came from the field of image recognition, where a naive representation of pictures is of a very high dimensionality, i.e., the number of pixels in the pictures. This can make the classification task prohibitly hard, unless an efficient method of dimensionality reduction is introduced. Low-dimensional representation was achieved by a "bottleneck" network (usually multi-layer perceptrons with 3 hidden layers) which was trained on related tasks.

In methods based on the construction of a *target-specific* concise representations, the representation is learned from a sample of labeled input instances. In such representations a considerable effort is devoted to the generation of a low dimensional representation of the input instances, which is generally suited only for the specific target concept. Representatives of this class are the activation pattern of the last "hidden layer" in a feed-forward multi-layer neural networks, boosting of "weak learners" (section 3.3) (where the set of weak learners that were chosen by the learning algorithm can be considered as the set of functions that are used for the vector representations) and learning with "ensemble" of neural networks, [166][127][91], where the output of each neural network in the ensemble can be considered as a component in the vector representation.

# Chapter 5

# Faithful Representations and Learning Algorithms

Faithful and balanced representations are esthetically appealing. However, the real merit of representations in learning is due to their impact on the learning process. In this chapter we will consider when, and to what extent, the use of faithful representations can simplify the learning process.

## 5.1 Linear separability using faithful representations

If, for an approximate faithful representation of $h_0$, using a random sample ${\mathcal{S}_{\mathcal{H}}}^m$ of $m$ hypotheses drawn from $\mathcal{H}$ according to $\mathcal{P}$,:

$$r_{\mathcal{S}_m}(x) = (h_1(x), \ldots, h_m(x))) \ ,$$

there exists a set of coefficients $\{\alpha_{0i}\}$ such that $\operatorname{sgn}\left(\sum_{i=1}^m \alpha_{0i} h_i(x)\right) = h_0(x)$, then the problem is said to be *linearly separable using a faithful representation.* As mentioned in section 2.5, linearly separable problems are considered to be relatively "easy," and many efficient learning algorithms for finding a set of coefficients that define the separating hyperplane exist. Although quite a few hypotheses spaces have this nice property, it can be shown[8] that, in general, most of the functions can not be approximated using such a simple scheme. However, even if the problem is not linearly separable using a faithful representation, an attempt to find a hypothesis $\hat{h} = \operatorname{sgn}\left(\sum_{i=1}^m \alpha_{0i} h_i(x)\right)$ which approximates $h_0$ is still very constructive: approximate hypotheses can serve as *weak learners* and standard "boosting" techniques can then be used in order to promote the final accuracy, as described in section 3.3. Moreover, unlike Boolean hypotheses, such hypotheses can output, apart from the mere sign of the input instance, also a "margin of confidence" for the sign of the input instance based on its distance from the separating hyperplane[177]. This property is most useful, since it allows for a special majority vote that can take into account an idiosyncratic level of confidence in the label of a given instance, and for a definition of a measure for "informative examples."

Since a faithful representation scheme transforms the hypotheses to vectors in $R^n$, the simplest scheme for dichotomy learning is first to attempt to *approximate* the representation of the target hypothesis, $r_{S_n}(h_0) = \mathbf{h}_0$, using a linear combination of the vectors that corresponds to the representations of the hypotheses in the hypotheses sample:

$$\hat{r}(h) = \sum_{i=1}^{m} \alpha_{0i} r(h_i). \tag{5.1}$$

And then to define a binary hypothesis $\hat{h_0}$ that approximate the target hypothesis:

$$\hat{h_0}(x) = \operatorname{sgn}\left(\sum_{i=1}^{m} \alpha_{0i} h_i(x)\right). \tag{5.2}$$

Another way to view this problem is as a Bayesian decision problem[43]: denoting $\mathcal{X}_+ = \{x|h_0(x) = 1\}$ and $\mathcal{X}_- = \{x|h_0(x) = -1\}$, we define two world states: $\omega_1$, in which an input instance is drawn from $\mathcal{X}_+$ according to the restriction of $\mathcal{D}$ to $\mathcal{X}_+$, and $\omega_2$, in which an input instance is drawn from $\mathcal{X}_-$ according to the restriction of $\mathcal{D}$ to $\mathcal{X}_-$. Given an input instance $x'$, a Bayes-optimal procedure would be to assign it a label:

$$\operatorname{sgn}\left(P(\omega_1|x') - P(\omega_2|x')\right) = \operatorname{sgn}(g(x')), \tag{5.3}$$

where $g(x)$ is the Bayes discriminant function. A simple approximation for $g(x)$ using the faithful representation scheme is the series expansion:

$$g(x) = \sum_{i=1}^{m} \alpha_{0i} h_i(x)$$

which gives equation 5.3 the same form as equation 5.2.

Different criteria for approximation would yield different learning procedures.

### 5.1.1 Local learning algorithms using faithful representations

The first type of conditions considered here are *local conditions*, where each of the weights that define the separating hyperplane can be evaluated independently from the other weights. Local algorithms are studied here for three main reasons:

1. The computational complexity of local algorithms is proportional to $n \cdot m$, where $n$ is the number of input samples in the training set and $m$ is the number of hypotheses that were sampled. This computational complexity is smaller than the (worst-case) computational complexity of non-local algorithm using the same setting, and allows for an efficient parallelization.
2. The known learning processes in biological systems are local in nature, and therefore local learning algorithms can give some insight for learning processes in biological systems.
3. Local algorithms are inherently robust to noise.

### Finding a Separating Hyperplane Using a Hebb-like Algorithm

Hebb's postulate of learning is the oldest and most famous learning rule: quoting from Hebb's book, "The organization of behavior" (1949, p 62)

"When an axon of a cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic changes take place in one or both cells such that A's efficiency as one of the cells firing B, is increased."

In the context of machine learning, Hebb-like rules are rules in which the assigned weights between two units are proportional to the correlations between their outputs.

Given a sample of $n$ labeled input instances, $((x_1, h_0(x_1)), \ldots, (x_{m_x}, h_0(x_n)))$ and a representation of the input instances using a sample of $m$ hypotheses $h_1, \ldots, h_m$, we define the output of the Hebb-like algorithm as:

$$
\begin{aligned}
\hat{h}_0(x) &= \operatorname{sgn}\left[\frac{1}{n \cdot m}\sum_{i=1}^{n}\sum_{j=1}^{m} h_0(x_i)h_j(x_i)h_j(x)\right] \\
&= \operatorname{sgn}\left[\frac{1}{n \cdot m}\sum_{j=1}^{m} (\mathbf{h}_0 \cdot \mathbf{h}_j)\, h_j(x)\right] \\
&= \operatorname{sgn}\left[\frac{1}{n \cdot m}\sum_{i=1}^{n} h_0(x_i)\, (\mathbf{x}_i \cdot \mathbf{x})\right] ,
\end{aligned}
\tag{5.4}
$$

*where* $\mathbf{a} \cdot \mathbf{b}$ *denotes the dot (scalar) product, and the representation vectors are defined as* $\mathbf{h} = (h(x_1), ..., h(x_n)) \in \mathbf{R}^n$ *and* $\mathbf{x} = (h_1(x), ..., h_m(x)) \in \mathbf{R}^m$ *respectively.*

We refer to the first form in Eq.(5.4) as the *direct representation* and the second as the *dual representation* of the label $\hat{h}_0(x)$. A nice property of the Hebb algorithm is that the two representations have the same form and analysis of its performance can be obtained using the most convenient representation.

The Hebb algorithm can be viewed as a method for constructing un-normalized *centroids*, or centers of gravity, for the representations of the negative and positive examples:

$$
\begin{aligned}
\bar{\mathbf{x}}_+ &= \sum_{h(x_i)=+1} \mathbf{x}_i, \\
\bar{\mathbf{x}}_- &= \sum_{h(x_i)=-1} \mathbf{x}_i,
\end{aligned}
\tag{5.5}
$$

and assign the label of a novel input instance by comparing its scalar product with each centroid.

If there are no correlations between the outputs of the hypotheses in the space, i.e., if the representations of any two hypotheses are orthogonal, then all the values of all the weights are random and the Hebb algorithm is of no use. Such is the case for the hypotheses space of partial parity functions. The Hebb algorithm can be successfully applied only if there is a sufficient amount of correlations between the hypotheses in the space. We now study sufficient conditions for the applicability of the Hebb algorithm and bounds on the size of hypotheses and input instances samples.

### Evaluating the asymptotic error of the Hebb-like algorithm

Denoting: $\alpha_{ij} = \frac{1}{n}\mathbf{h}_i \cdot \mathbf{h}_j$, equation (5.4) can be written as:

$$\hat{h}_0(x) = \operatorname{sgn}\left[\frac{1}{m}\sum_{j=1}^{m} \alpha_{0j} h_j(x)\right]. \tag{5.6}$$

Denoting $\eta(h_i, h_j) = E_x(h_i(x)h_j(x))$, and

$$\xi(x) = \frac{h_0(x)\sum_{j=1}^{m} \alpha_{0j} h_j(x)}{\sum_{j=1}^{m} \alpha_{0j}^2},$$

then, since $\hat{h}_0(x) \neq h_0(x)$ if, and only if, $h(x)\sum_{j=1}^{m} \alpha_{0j} h_j(x) < 0$, the error of the Hebb-like algorithm is the expected number of $x'$s for which $\xi(x) < 0$.

The *expectation value* of $\xi(x)$ is:

$$\begin{aligned} E_x(\xi(x)) &= E_x\left(\frac{\sum_{j=1}^{m} \alpha_{0j} h_j(x) h_0(x)}{\sum_{j=1}^{m} \alpha_{0j}^2}\right) \\ &= \frac{\sum_{j=1}^{m} \alpha_{0j}\eta(h_0, h_j)}{\sum_{j=1}^{m} \alpha_{0j}^2}. \end{aligned} \tag{5.7}$$

Denoting $\epsilon_{ij} = \eta(h_i, h_j) - \alpha_{ij}$ and $\langle \alpha_0^2 \rangle = \frac{1}{m}\sum_{j=1}^{m} \alpha_{0j}^2$, this can be written as:

$$\begin{aligned} E_x(\xi(x)) &= \frac{\sum_{j=1}^{m} \alpha_{0j}(\alpha_{0j} + \epsilon_{j0})}{\sum_{j=1}^{m} \alpha_{0j}^2} \\ &= 1 + \frac{\sum_{j=1}^{m} \alpha_{0j}\epsilon_{j0}}{m\langle \alpha_0^2 \rangle}. \end{aligned} \tag{5.8}$$

Denoting $p_{ij} = \mathbf{Pr}_x(h_i(x) = h_j(x))$ and by $\hat{p}_{ij}$ the relative number of times in which $h_i(x_k) = h_j(x_k)$, $k = 1 \ldots n$, we have $\eta(h_i, h_j) = 2p_{ij} - 1$, $\alpha_{ij} = 2\hat{p}_{ij} - 1$ and $\epsilon_{ij} = 2(p_{ij} - \hat{p}_{ij})$. $\epsilon_{ij}$ is therefore a random variable that corresponds to the difference between the expected number of successes and the actual frequency of successes in $n$ independent Bernoulli trials, and therefore $\epsilon_{ij}$ is distributed binomially with zero mean and variance of $\sqrt{\frac{p_{ij}(1-p_{ij})}{n}}$. Since $\alpha_{ij} \leq 1$ and $\alpha_{ij}$ is not correlated with $\epsilon_{ij}$, we have $\sum_{j=1}^{m} \alpha_{0j}\epsilon_{j0} = O(\sqrt{m}\cdot 1/\sqrt{n})$ and therefore:

$$E_x(\xi(x)) = 1 + \frac{O(\sqrt{m}\cdot 1/\sqrt{n})}{m\langle \alpha_0^2 \rangle} \tag{5.9}$$

and as $n$ tends to $\infty$, $E_x(\xi(x))$ tends to 1.

In order to bound the probability of error, $\mathbf{Pr}_x(\xi(x) < 0)$, we need to evaluate the *variance* of $\xi(x)$. The second moment of $\xi(x)$ is:

$$
\begin{aligned}
E_x(\xi^2(x)) &= E_x\left(\frac{\left(h_0(x)\sum_{j=1}^m \alpha_{0j}h_j(x)\right)^2}{\left(\sum_{j=1}^m \alpha_{0j}^2\right)^2}\right) \\
&= \frac{\sum_{j=1}^m \alpha_{0j}^2}{(m\langle\alpha_0^2\rangle)^2} + \frac{2\sum_{i<j}^m \alpha_{0i}\alpha_{0j}\eta(h_i,h_j)}{\left(\sum_{j=1}^m \alpha_{0j}^2\right)^2} \\
&= \frac{1}{(m\langle\alpha_0^2\rangle)} + \frac{2\sum_{i<j}^m \alpha_{0i}\alpha_{0j}\eta(h_i,h_j)}{(m\langle\alpha_0^2\rangle)^2}
\end{aligned}
\tag{5.10}
$$

In the limit $n, m \to \infty$. $\alpha_{ij} = \eta(h_i, h_j)$. Denote the "metric variance" of hypotheses in $\mathcal{H}$ with respect to $h_0$ by

$$\Delta_0 = \left(E_{h\in\mathcal{H}}(\eta(h_0,h)^2)\right)^2$$

and the average "three-point correlation" by

$$\Delta_1 = E_{h',h''\in\mathcal{H}}(\eta(h_0,h')\eta(h_0,h'')\eta(h',h''))).$$

If $\Delta_0 > \epsilon > 0$ then, neglecting terms of order $1/m$ equation (5.10) can be written as:

$$E_x(\xi^2(x)) = \frac{\Delta_1}{\Delta_0}. \tag{5.11}$$

and the variance of $\xi$ is:

$$Var(\xi) = E_x(\xi^2(x)) - E_x^2(\xi(x)) = \frac{\Delta_1}{\Delta_0} - 1. \tag{5.12}$$

Using the Chebyshev inequality, and the fact the $E_x(\xi) = 1$, the probability of error of the Hebb-like algorithm can be bounded by:

$$\mathbf{Pr}(error_B) = \mathbf{Pr}(\xi < 0) \leq \frac{\Delta_1}{\Delta_0} - 1. \tag{5.13}$$

And the error's probability goes to zero if:

$$E_{h',h''\in\mathcal{H}}(\eta(h_0,h')\eta(h_0,h'')\eta(h',h'')) = E_{h'\in\mathcal{H}}(\eta(h,h')^2). \tag{5.14}$$

From the above analysis one can see that the probability of error reduces if the hypotheses in the sample have, on the average, large correlations with the target hypothesis, (i.e., $\Delta_0$ is large) and small correlations between themselves (which tends to reduce $\Delta_1$). Within the faithful representation framework, the error of Hebbian algorithms remain in general finite. However, it is very easy and computationally efficient to apply the ADABOOST algorithm, described in section 3.3, for a Hebbian learner: the probabilistic weights $\mathbf{P^t}$ in equation 3.6 define a metric for the scalar product between two hypotheses, so the scalar product is defined as:

$$\mathbf{h}_i \cdot \mathbf{h}_j = \sum_{k=1}^{n} P^t(x_k) h_i(x_k) h_j(x_k) \tag{5.15}$$

which make a full use in the entire sample.

Since in the Hebbian scheme there is a complete symmetry between the roles of the hypotheses and the input instances, the same analysis holds also if we replace the roles of the hypotheses and input instances in the above analysis. In many cases, before using any representational shift, the input instances are vectors in $R^n$, while the hypotheses have more complicated structure (e.g., decision trees, multi-layer perceptrons etc..) and therefore analyzing the performance of the Hebb-like algorithm can be easier using the "input domain".

### 5.1.2 Least square approximation

Finding the candidate separating hyperplane using a least square approximation method is computationally efficient, easy to analyze and often times yields rather good results.

Given a sample of $n$ labeled input instances, independently drawn from $\mathcal{X}$ according to a common distribution $\mathcal{D}$,
$S_n = ((x_1, h_0(x_1)), \ldots, (x_n, h_0(x_n)))$, and a sample of $m$ hypotheses $h_1, \ldots, h_m$ independently drawn from $\mathcal{H}$ according to $\mathcal{P}$, the vector representation of $h_0$, $\mathbf{h}_0$, is given by $\mathbf{h}_o = (h_0(x_1), \ldots, h_0(x_n))$, and the vector representation of $h_1, \ldots, h_m$ is given by $\mathbf{h}_i = (h_i(x_1), \ldots, h_i(x_n))$. We attempt to find a vector $\hat{h_0} \in R^n$, $\hat{\mathbf{h}_0} = \sum_{i=1}^{m} \alpha_{0i} \mathbf{h}_i$ such that $||\hat{\mathbf{h}} - \mathbf{h}_0||^2$ is minimal, and then define

$$\hat{h_0}(x) = \mathrm{sgn}\left(\sum_{i=1}^{m} \alpha_{0i} h_i(x)\right) \tag{5.16}$$

This method is well known (see, for example, Duda & Hart, pp. 151-159[43]). Two widely used methods for evaluating the set of coefficients, $\alpha_{01}, \ldots, \alpha_{0m}$, the *pseudo-inverse method* and the *Widrow-Hoff algorithm,* are described below.

1. **The Pseudo-Inverse method** Let $\mathbf{A}$ be the n-by-m matrix whose raws are vector representations of input samples in $R^m$ and whose columns are vector representations of hypotheses samples in $R^n$:

$$\mathbf{A} = \begin{pmatrix} h_1(x_1) & h_2(x_1) & \ldots & h_m(x_1) \\ h_1(x_2) & h_2(x_2) & \ldots & h_m(x_2) \\ \cdots & \cdots & \cdots & \cdots \\ h_1(x_n) & h_2(x_n) & \ldots & h_m(x_n) \end{pmatrix},$$

The problem is to find a weight vector $\alpha$ s.t. $\mathbf{J}(\alpha) = ||\mathbf{A}\alpha - \mathbf{h}_0||^2$ is minimal. A simple closed form solution can be found by forming the gradient:

$$\nabla \mathbf{J} = 2\mathbf{A}^t(\mathbf{A} \cdot \alpha - \mathbf{h}_0), \tag{5.17}$$

and setting it equal to zero. This yields the necessary condition:

$$\mathbf{A}^t\mathbf{A} \cdot \alpha = \mathbf{A}^t\mathbf{h}_0. \tag{5.18}$$

If $\mathbf{A}^t\mathbf{A}$ is non-singular, there exists a unique solution for $\alpha$:

$$\alpha = (\mathbf{A}^t\mathbf{A})^{-1}\mathbf{A}^t\mathbf{h}_0. \tag{5.19}$$

and the matrix $\mathbf{A}^+ = (\mathbf{A^tA})^{\mathbf{-1}}\mathbf{A^t}$ is called the *pseudo-inverse* of $\mathbf{A}$.

If $\mathbf{A}^t\mathbf{A}$ is not singular, then the solution to equation (5.18) is not unique. However, a minimum-square-error solution always exists. In particular, $\mathbf{A}^+$ can be defined more generally as:

$$\mathbf{A}^+ = \lim_{\epsilon \to 0}(\mathbf{A}^t\mathbf{A} + \epsilon I)^{-1}\mathbf{A}^t, \tag{5.20}$$

it can be shown that this limit always exists, and that $\alpha = \mathbf{A}^+\mathbf{h}_0$ is a minimum square error solution to $\mathbf{A}\alpha = \mathbf{h}_0$.

The pseudo-inverse of the matrix $\mathbf{A}$ can be defined in terms of its singular-value-decomposition (SVD),

$$\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^{^T} \Longrightarrow \mathbf{A}^+ = \mathbf{V}\mathbf{S}^+\mathbf{U}^{^T} \tag{5.21}$$

where $\mathbf{V}$ is the orthonormal matrix of the eigenvectors of $\mathbf{A}^{^T}\mathbf{A}$ and $\mathbf{U}$ is the orthonormal matrix of the eigenvectors of $\mathbf{A}\mathbf{A}^{^T}$. $\mathbf{S}$ is the diagonal matrix consisting of the singular values of $\mathbf{A}$ while $\mathbf{S}^+$ is the diagonal matrix consisting of the inverse singular values of $\mathbf{A}$, for those values that are greater than some threshold $\epsilon_{_T}$. Notice that $\mathbf{S^2}$ is the diagonal matrix consisting of eigenvalues of both of $\mathbf{A}\mathbf{A}^{^T}$ and $\mathbf{A}^{^T}\mathbf{A}$.

This scheme is particularly useful for multi-task learning, where several target concepts taken from $\mathcal{H}$ are needed to be learned: if another target $h'$ is needed to be learned using the same sample $S_n$, all that is required is to multiply its representation by the same pseudo-inverse, $\mathbf{A}^+$,

$$\alpha' = \mathbf{t}' \cdot \mathbf{A}^+.$$

2. **Gradient descent and Widrow-Hoff method**

$\mathbf{J}(\alpha) = ||\mathbf{A} \cdot \alpha - \mathbf{h}_0||^2$ can also be minimized using the gradient descent procedure. Since $\nabla J = 2\mathbf{A}^t(\mathbf{A} \cdot \alpha - \mathbf{h}_0)$, an obvious gradient descent method is:

- choose an arbitrary $\alpha_1$

- Set $\alpha_{k+1} = \alpha_k - \rho_k \mathbf{A}^t(\mathbf{A} \cdot \alpha_{\mathbf{k}} - \mathbf{h}_0)$

It can be shown that if

$$\rho_k = \frac{\rho_1}{k},$$

then the sequence of weights $\{\alpha_k\}$ converges to a limiting vector $\alpha$ satisfying:

$$\mathbf{A}^t(\mathbf{A} \cdot \alpha - \mathbf{h}_0) = 0$$

Thus the gradient descent algorithm always yields a solution, even if $\mathbf{A}^t\mathbf{A}$ is singular.

This method still requires the storage of the $m \times m$ matrix $\mathbf{A}^t\mathbf{A}$. The *Widrow-Hoff* method allows for a further reduction in the storage requirement, by taking at each step the representation of one input instance, $r_{\mathcal{S}_m}(x_k) = (h_1(x_k), \ldots, h_m(x_k)))$ and using the following rule:

- choose an arbitrary $\alpha_1$
- set

$$\alpha_{k+1} = \alpha_k + \rho_k \left(\mathbf{h}_0(x_k) - \alpha_{\mathbf{k}}^{\mathbf{t}} r_{\mathcal{S}_m}(x_k)\right) r_{\mathcal{S}_m}(x_k)$$

The Widrow-Hoff rule corrects the coefficient vector $\alpha$ whenever the product $\alpha_{\mathbf{k}}^{\mathbf{t}} r_{\mathcal{S}_m}(x_k)$ does not equal $\mathbf{h}_0(x_k)$. It can be shown that the weight vector $\alpha$ tends to desired LMS solution.

### 5.1.3 Analysis of the pseudo-inverse algorithm

The pseudo-inverse method for solving a system of linear equations finds the projection of $r(h_t)$ into the space: $\operatorname{span}(r(h_1), \ldots r(h_m))$

$$\tilde{r}(h_t) = \sum_{i=1}^{m} \alpha_i r(h_i)$$

The generalization error of the output hypothesis

$$\tilde{h}(x) = \operatorname{sgn}\left(\sum_{i=1}^{m} \alpha_i(h) \cdot h_i(x)\right)$$

can be easily bounded by[92]:

$$\mathbf{Pr}\left[h_t(x) \neq \tilde{h}(x)\right] \leq E_x\left[\left(h(x) - \sum_{i=1}^{m} \alpha_i(h) \cdot h_i(x)\right)^2\right]. \tag{5.22}$$

The generalization error is therefore bounded by the square length of the projection of $r(h_t)$ on the subspace orthogonal to $\operatorname{span}(r(h_1), \ldots r(h_m))$.

### Principal components and spectrum of a Boolean learning problem

To bound the error of this learning algorithm we need an abstract construction, very familiar in other contexts, called *principal components of the learning problem.* The use of this construction is mainly for the sake of simplicity and clarity of the arguments. In practice we deal only with approximations to this construction.

Given a learning problem $\{\mathcal{H}, \mathcal{P}, \mathcal{X}, \mathcal{D}\}$, when the VC-dimensions of both the primal and the dual learning problems are finite, it is always possible to construct a finite *$\epsilon$-cover* for both $\mathcal{H}$ and $\mathcal{X}$, (namely, finite subsets $\mathcal{H}_\epsilon \subset \mathcal{H}$ and $\mathcal{X}_\epsilon \subset \mathcal{X}$ such that for any $h \in \mathcal{H}$ there is an $h' \in \mathcal{H}_\epsilon$ s.t. $\epsilon_g(h, h') < \epsilon$ and similarly for $x \in \mathcal{X}$).

Given finite $\epsilon$-covers of $\mathcal{H}$ and $\mathcal{X}$, it is possible to consider the finite matrix $\mathbf{A}^0$ s.t. $a^0_{ij} = h_i(x_j)$ are the labels of the set $\mathcal{H}_\epsilon$ on the set $\mathcal{X}_\epsilon$.

Consider the symmetric positive *correlation matrix* $\mathbf{A}_0\mathbf{A_0^T}$. The eigenvectors of $\mathbf{A}_0\mathbf{A_0^T}$ define the principal components (PC) of $\mathcal{H}_\epsilon$, while the eigenvectors of $\mathbf{A_0^T}\mathbf{A_0}$ are the PC of $\mathcal{X}_\epsilon$. We refer to the the raws of $\mathbf{A}_0$ as *the full vector representation of the the hypotheses in* $\mathcal{H}_\epsilon$ and to the columns of $\mathbf{A}_0$ as *the full vector representation of the the inputs in* $\mathcal{X}_\epsilon$

The corresponding eigenvalues (principal values) that are identical for both matrices are denoted by $\sigma_1^2, \sigma_2^2 ...$ and referred as *the spectrum of the learning problem.* There are several important observations about the spectrum.

1. The spectrum is normalized: $\sum \sigma_i^2 = 1$.
2. The dual learning problem has the same spectrum as the primal learning problem. This means that the diagonal form of the covariance matrices of the the two problems are equal. If the distribution of the distances between the representations of hypotheses and input instances are Gaussian, this would mean that the two distributions are essentially equal.
3. The spectrum depends on the distributions $\mathcal{P}$ and $\mathcal{D}$ but, due to the uniform convergence property of finite VC-dimension spaces, is not sensitive to the specific subsets $\mathcal{H}_\epsilon$ and $\mathcal{X}_\epsilon$.
4. The principal components can also be derived using an Hebbian algorithm[152], and so the above algorithm can be made biologically plausible.

Fig. 5.1 shows the characteristic spectrum of several learning problems, where $\mathcal{P}$ and $\mathcal{D}$ are uniform distributions over the hypotheses and over the input space, respectively. (experimental study of several learning problems will be discussed in chapter 5)

In particular, if most of the "weight" the spectrum is concentrated in a small number of principal values, the problem can be approximated by a projection into the subspace of these components. This allows for a faithful *dimensionality reduction.* We name such hypotheses spaces *informative.* Formally:

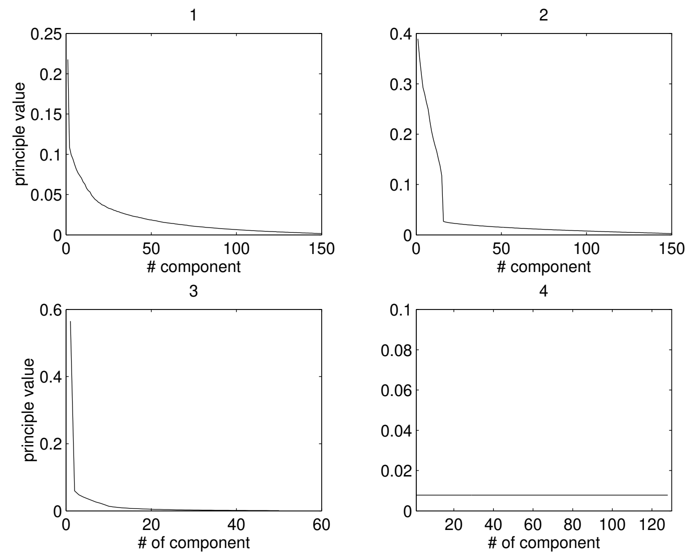


Figure 5.1: Principal value spectrum for (1) balanced decision trees of depth 4 with fixed node variables. (2) 15 dimensional perceptrons. (3) 4-dimensional rectangles. (4) partial parity functions.

**Definition 3 Informative learning problem** *A learning problem* $\{(\mathcal{HP}), (\mathcal{X}, \mathcal{D})\}$ *is called* informative *if for any* $\epsilon, \delta > 0$ *there exists a number* $m_0$*, polynomial in* $\frac{1}{\epsilon}, VCd(\mathcal{H})$*, and* $VCd(\mathcal{X})$*, such that, with probability of at least* $1 - \delta$, $\sum_{i=m_0}^{rank(A)} \sigma_i^2 < \epsilon$.

Generally, it is very hard to tell weather a certain problem is informative, since the probability density of the spectrum of random matrices is known only for a small number of distributions. One notable example for which the probability density of the eigenvalues of a random matrix is known is the Gaussian case, in which the matrix elements are independently drawn according to a Gaussian distribution. In this case the eigenvalues are distributed according to Wigner's semi-circle law[183]:

$$P(x) = \begin{cases} (2/\pi)\sqrt{1-x^2} & \text{if } |x| \leq 1 \\ 0 & \text{otherwise} \end{cases} \tag{5.23}$$

This result can be extended to covariance matrices of set of vectors whose elements are normal variables[79][180]: let $x_{ij}$, $i = 1, \ldots, m$, $j = 1, \ldots, n$, be independent normal variables with zero mean and unit variance, and let $y_{rs} = \sum_{j=1}^{n} x_{rj} x_{sj}$ denote the elements of a $m \times m$ matrix, $\mathbf{S}_m^{(n)}$. The distribution of the matrix $\mathbf{S}_m^{(n)}$ is called central *Wishart* distribution with $n$ degrees of freedom.

In the limit $n, m \to \infty$, $\frac{m}{n} = \alpha$, the distribution function of the eigenvalues converge in probability to a distribution with density $P_\alpha(x)$. For $0 < \alpha \leq 1$, denote $a = 1 + \alpha - 2\sqrt{\alpha}$

and $b = 1 + \alpha + 2\sqrt{\alpha}$, this density has the form:

$$P_\alpha(x) = \begin{cases} \frac{\sqrt{(x-a)(b-x)}}{2\pi\alpha x} & \text{for } a < x < b \\ 0 & \text{otherwise} \end{cases} \tag{5.24}$$

for $1 < \alpha < \infty$ the probability density is a mixed density:

$$P_\alpha(x) = \begin{cases} \frac{\sqrt{(x-a)(b-x)}}{2\pi\alpha x} & \text{for } a < x < b,\ x \neq 0 \\ 1 - \frac{1}{\alpha} = 1 - \frac{n}{m} & \text{for } x = 0 \\ 0 & \text{otherwise} \end{cases} \tag{5.25}$$

This result also holds if the variables $x_{ij},\ i = 1, \ldots, m,\ j = 1, \ldots, n$ are $n \times m$ independent random $\pm 1$ variables, with $\Pr(\mathrm{x_{ij}} = +1) = \Pr(\mathrm{x_{ij}} = -1) = 1/2$ [36]. However, most hypotheses spaces that are encountered in practice induce rather complicated distributions of distances[1] and the spectrum can not be predicted theoretically.

## 5.2 Some experimental results

### 5.2.1 Learning balanced decision trees

As mentioned in section 3.4, decision trees provide a simple, yet extremely efficient method for classification and regression[12][142]. Each decision tree defines a boolean function. An assignment to the variables determines a unique path from the root to a leaf: at each internal node the left (right) edge to a child is taken if the value of the node variable in the assignment is 1 (0). The value of the tree function at the assignment is the value at the leaf reached. (see figure 3.1). The *depth* of a decision tree is the length of the longest path from the root to the a leaf. If all the paths have the same length, we say that the decision tree is *balanced*.

Most of the machine learning algorithms that use decision trees, employ an heuristics for building top-down decision trees from labeled sample data. Such algorithms grow a tree from the root to the leaves by repeatedly replacing an existing leaf with an internal node, thous "splitting" the data reaching this leaf into two new leaves, and thereby reducing the empirical error on the given sample. Various heuristics are differ mainly by the methods and the criteria for replacing leaves by nodes.

Although in principle, no efficient algorithm for learning decision trees exists, various heuristics for building top-down decision-trees give good performance when applied to real data. In a late study Mansour and Kearns[83] suggested that this may be due to the fact that the method by which top-down decision-trees are built from data is essentially a "boosting" method, where the nodes of the trees can be considered as "weak learners".

[1] One notable exception is the set of "partial parity functions" discussed in section 3.5, where the distance between the representations of each pair of input instances and each pair of hypotheses goes asymptoticly to 1/2.

In our study we considered the following challenging problem of decision-tree learning: the input instances are strings of $n$ boolean variables, $\mathbf{x} = (x_1, \ldots, x_n)$, the (unknown) target decision tree consist of one of the variables in each of the nodes, together with a label $\{1, 0\}$, and each variable is assigned to one, and only one, node and the tree is known to be balanced. Both the instances and the target tree are drawn according to a uniform distribution. This setting represent a "worst-case scenario", since the heuristics for building top-down decision-trees can not find a useful splitting criterion. In an easier version of this problem, the assignment of variables to nodes is known, and only the labels of the nodes are need to be found.

A faithful representation of the inputs in the "hard" problem is induced by selecting $m$ balanced decision trees with $n$ nodes, where the indexes and the signs of the node variables are drawn at random according to a uniform distribution.

Fig. 5.2 shows the results of two learning algorithms, that use the faithful representation for the "easy" problem of learning decision trees with 15 nodes: (i.e., when the identity of the node variables is known, and the various hypotheses are differ in the labels of the node variables) the upper curve represents the results of the pseudo-inverse algorithm, described in section 5.1.2, and the solid curve – the "AdaBoost" algorithm, which uses the Hebb learning algorithm as a "weak-learning" method, as described in section 5.1.1. The $x$-axis described the number of random hypotheses which were used for the representation and the $y$-axis – the generalization error. 2000 examples were used for training and for testing the generalization error in each of the algorithms.

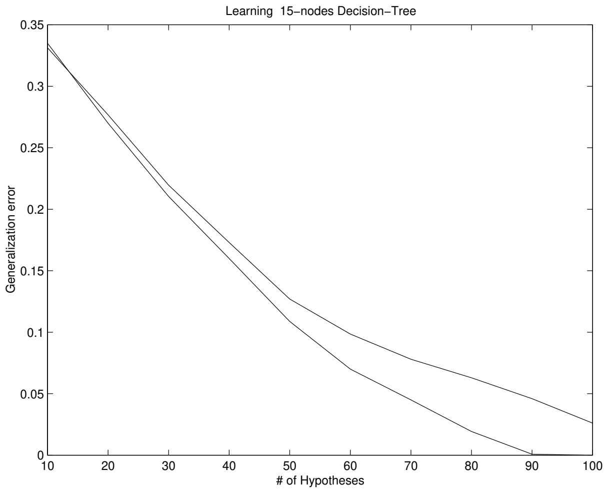


Figure 5.2: Learning "easy" 15-nodes decision-trees using the pseudo-inverse method (lower curve) and the "boosted" Hebbian method (upper curve)

Fig. 5.3 shows the results of the same algorithms for the "hard" learning problem. 1800 examples were used for training and for testing the generalization error in each of the algorithms.

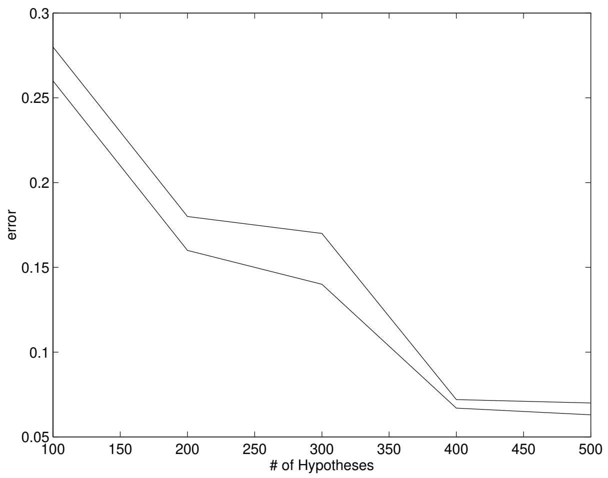


Figure 5.3: Learning "hard" 15-nodes decision-trees using the pseudo-inverse method (lower curve) and the "boosted" Hebbian method (upper curve)

### 5.2.2 Learning of high-dimensional rectangles

Learning of rectangles serves as a canonical example in computational learning theory. In real-life applications, rectangles can be considered as a scheme for learning a concept which is a set of independent features, each one is bounded between two values. e.g., the concept of "a man with a medium built" can be defined as a man whose weight is between 170-180 cm. and is weight is between 65-75 kg. The faithful representation of the input instances is the vector of the labels, given to that input instance by a set of $m$ random d-dimensional rectangles, drawn according to some probability distribution $\mathcal{P}$ over the hypotheses space. In the following experiments, we considered a set of d-dimensional rectangles in the d-dimensional unit cube that contained the center of the cube. The rectangles were defined using two set of coordinates: $x_{\text{min},1}, \ldots, x_{\text{min},d}$ and $x_{\text{max},1}, \ldots, x_{\text{max},d}$, where $x_{\text{min},i}$ are distributed uniformly in $[0, \alpha]$ and $x_{\text{max},i}$ are distributed uniformly in $[1-\alpha, 1]$, and $\alpha$ was chosen such that the average volume would be $1/2$. The pseudo-inverse algorithm gave an excellent solution to that problem. Fig. 5.4 shows the generalization error of the algorithm as a function of the number of hypotheses used for the representation. The algorithm use 400 samples for training and generalization and the results are averaged over 100 cases.

## 5.3 Summary of part II

In this part of the work we introduce a setting for representation and learning that uses the idiosyncratic feature of the learning problem, while remains general enough to be implemented on many different problem classes. We consider the Hamming distance between the constructed representations of any two hypotheses, and show that this distance approximates the probability of their disagreement with respect to the labels of input instances (i.e.,

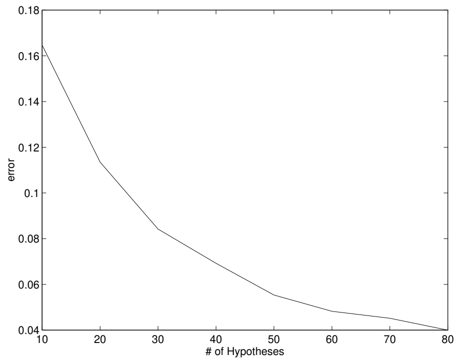


Figure 5.4: Learning 5-dimensional rectangles using the pseudo-inverse method

the relative generalization error of one hypothesis w.r.t. the other). Using a "dual learning problem" setting, we also show that the Hamming distance between the representation of any two input instances approximates the probability that a random hypothesis will not assign the same label to the two inputs. The representations are balanced in the sense that all the coordinates in the representation are statistically equal. Furthermore, the fact that the representations are binary allow efficient evaluation of distances and correlations between hypotheses and between input instances. We discussed two basic methods that make use of faithful and balanced representations. The first applies a "local" algorithm that use the correlations between the representations of the target hypotheses and the representations of a pre-defined set of hypotheses. This setting is inspired from a basic biological learning mechanism, known as Hebb's rule. This algorithm, in general, does not produced good results, but it can be "boosted" in an efficient manner, using the "AdaBoost" algorithm of Freund and Schapire. The second employs least-mean-square approximators that attempt to approximate a separating hyperplane between the positive and the negative examples. Results of simulations of the algorithms gave good results for the challenging problem of learning Boolean decision trees under a uniform distribution.

# Part III

# Learning and Constraint Satisfaction

# Chapter 6

# The dual problem of CNF learning and the k-SAT problem

*"Knowing how to be satisfied is granted to bring satisfaction"*

*Lao-Tse, "Tao Teh Ching", 46*

## 6.1 Learning as a constraint satisfaction problem

Given a hypotheses space $\mathcal{H}$ and a set of labeled input samples, $((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m))$, a learning problem can be considered as a *constraint satisfaction problem* in the following sense: each labeled input instance can be viewed as a constraint w.r.t. the hypotheses space, and the goal is to find a hypothesis $h_0 \in \mathcal{H}$ that satisfies the maximal number of constraints. As discussed in the introduction, under appropriate conditions, solving this problem can assure us that we would find a hypothesis that generalizes well.

From the computational complexity point of view, many constraint satisfaction problems belong to the NP-complete class. For such problems, for any known algorithm, it is always possible to find instances whose resolution required computational resources that scale exponentially with the size of the inputs. This, however, does not say much about the *typical computational complexity*, namely, the number of computational steps needed for the resolution of a typical instance.

As a prototype of a constraint satisfaction problem, the dual problem of CNF learning from positive examples is studied, and its typical *sample complexity* is analyzed. In this problem, one attempts to find an assignment satisfying the CNF expression that is generated by the Boolean product of the sample, as described in section 6.2. Finding such an assignment is equivalent to solving the problem of satisfying k-CNF formulae (the k-SAT problem). This problem is a generic computational problem that became a cornerstone in the theory of computational complexity. In addition, k-CNF formulae constitute a general paradigm for Boolean constraints satisfaction and as such are of tremendous practical interest. k-CNF's appear naturally in a variety of applications, e.g., program and machine testing, VLSI design, logic programming, Boolean inference, machine learning, etc. Therefore, considerable attention has been devoted to the statistical properties of random k-CNF formulae that are likely to appear with an efficient encoding of the problem (e.g., [19],[1][29][30][35][40] [41][51][82][86][88][90][109][110] [111][112][113]). Results regarding this general problem are described in chapter 7.

The organization of this chapter is as follows: In section 6.2 I describe the general notions of CNF formula and the dual CNF learning problem. Section 6.3 analyzes the sample complexity of the problem. Specifically, it addresses the following question: given an unknown assignment of $N$ Boolean variables, how many randomly selected CNF expressions, that are known to be satisfied by that assignment, are needed in order to correctly predict, with high probability, whether a new CNF expression is satisfied by the unknown assignment? Section 6.4 describes a novel algorithm that attempts to minimize the number of un-satisfied constraints (an "energy minimization" algorithm). This algorithm can find, with a high probability, a hypothesis that is consistent with the sample, i.e., a satisfying assignment to a k-CNF formula. The performance of this algorithm, for large number of variables, was found to be better than other published results.

While in the worst cases finding such an assignment requires an exponential number of computational steps, it turns out that in most cases the actual number of computational steps that are required in order to obtain a solution, or to prove that no solution exists, scales polynomially with the size of the problem. This feature, as well as other interesting features of the k-SAT problem, is discussed in chapter 7.

## 6.2 Preliminaries

A k-Conjunctive-Normal-Form (k-CNF) formula of the $N$ boolean variables $x_1, x_2, \ldots, x_N$ is a boolean formula made of a conjunction of $M$ disjunctive clauses,each containing precisely $k$ of the $N$ variables or their negation. (both $x_i$ and $\neg x_i$ are referred as *literals*)

$$F(x_1 \ldots x_N) = \overbrace{(\bar{x}_a \vee \cdots \vee \overbrace{x_c}^{literal}) \wedge \cdots \wedge \underbrace{(x_d \vee \cdots \vee x_f)}_{clause}}^{M-clauses}$$

A satisfying assignment to the formula is an assignment of the $N$ variables in $\{-1, 1\}^N$ that yields a *true* value of the formula. Every Boolean formula can be expressed as a k-CNF formula.

Learning CNF formulae from a sample of labeled assignments (i.e., each assignment in the sample is labeled $+1$ if the assignment satisfies the target formula and $-1$ otherwise) is a well known problem in learning theory [85], and it is known to be efficiently PAC learnable. In the dual CNF learning problem, the hypotheses are assignments of $N$ boolean variables $x_1, x_2, \ldots, x_N$ in $\{-1, 1\}^N$, the target concept is a specific assignment, $\mathbf{x}_t$ and the input instances are disjunctive clauses of the form $(x_i \vee \neg x_j \vee x_l \ldots)$ together with a label that is $+1$

if $\mathbf{x}_t$ satisfy the clause (i.e. if $(x_{ti} \vee \neg x_{tj} \vee x_{tl} ...) = 1$) and $-1$ otherwise. This corresponds to a situation where there is no direct access to the variables themselves, and each time a test is made over a set of variables, such that a positive result is obtained if at least one of the variables in the test has an appropriate value. Since a result of most tests can have several interpretations, this reflects a rather general situation.

Formally, the problem can be described as follows:

- The target concept is an unknown assignment $\mathbf{x}_t \in \{-1, +1\}^N$.
- The input instances are disjunctive clauses of $k$ literals of the form: $(x_i \vee \neg x_j \vee x_l ...)$. The literals and their signs are drawn according to a uniform distribution: i.e. each one of the $N$ variables $x_1, \ldots, x_N$ has the same probability to be assigned to any given clause, and the sign of the variables in the clauses has a probability of 1/2 to be either $+1$ or $-1$.
- The learning algorithm gets $M$ labeled input instances, where the label is $+1$ if the clause is satisfied by $\mathbf{x}_t$ and $-1$ otherwise. The algorithm is then required to predict the label of a new input clause with error of less than $\epsilon$ and a confidence level of $1 - \delta$.

If a clause is labeled "negative," all the variables in the clause are completely determined. We therefore consider the more interesting case in which all the examples are positive, i.e., the knowledge about the target string is given in terms of a set of disjunctive clauses of k literals. This situation is common in many AI systems for representing knowledge using rules[144]. In this case, the $M$ positive examples determine a $k-$CNF formula. The space of all the consistent hypotheses – the version space – is the space of all satisfying assignments.[1]

## 6.3 The average sample complexity of the dual k-CNF learning problem

Without loss of generality we can study the case in which the target is $\mathbf{x}_t = (1, 1, \ldots, 1)$. Let $\{C_+\}$ denote the set of all k-clauses that $\mathbf{x}_t$ satisfy (i.e., all the k-clauses that do not contain a negation of all the k variables). Let $\{X_d\}$ denote the set of all the hypotheses that contain $d \cdot N$ variables whose value is $-1$, $0 \le d \le 1$ (i.e., all the assignments whose relative Hamming distance from $\mathbf{x}_t$ is $d$) and denote: $\gamma_1 = (1 - \frac{1}{2^k - 1})$. Using simple counting arguments, the probability that a hypothesis $x_i$, drawn at random according to a uniform distribution from $\{X_d\}$, would err on a given clause, $C'$, drawn at random according to a uniform distribution from $\{C_+\}$, is

$$P(\text{error}|d) = (1 - (1 - d)^k)(1 - \gamma_1). \tag{6.1}$$

[1] Another interesting situation is the case where both negative and positive examples exist, but there is a "labeling noise" in the negative examples, such that, with probability $\eta$, the label of a negative example may be wrong. In this case the positive examples determine a k-CNF formula and the negative examples provide a prior with regard to the variables that participate in the corresponding clauses, i.e., a prior over the space of satisfying assignments.

The probability of having a hypothesis with a Hamming distance $d$ in the version space can be evaluated using an analysis quite similar to the one we made in [172]. The probability that $\mathbf{x} \in \{X_d\}$ will be in the version space is:

$$\sum_{m=1}^{M} \binom{M}{m} P_0(d)^m (1 - P_0(d))^{(M-m)} \gamma_1^m = \left(1 - \frac{P_0(d)}{2^k - 1}\right)^M \quad . \tag{6.2}$$

Since there are $\binom{N}{d \cdot N}$ assignments with Hamming distance $d$ from $\mathbf{x}_t$, the probability to find a hypothesis in the version space whose distance from $\mathbf{x}_t$ is $d$ is:

$$Pr(\mathbf{x}' \in VS | d(\mathbf{x}', \mathbf{x}_t) = d) = \frac{1}{2^N} \binom{n}{d \cdot n} \left(1 - \frac{P_0(d)}{2^k - 1}\right)^M \quad , \tag{6.3}$$

We denote

$$F(\alpha, d, k) \equiv H(d) + \alpha \log_2 (1 - \frac{1 - (1-d)^k}{2^k - 1}) \ , \tag{6.4}$$

where $H(d)$ is the binary entropy,

$$H(d) \equiv -d \log_2(d) - (1-d) \log_2(1-d).$$

For large $N$ and $M \equiv \alpha N$, equation 6.3 can be written, using the Stirling approximation, as

$$\frac{1}{\sqrt{Nd(1-d)}} 2^{NF(\alpha,d,k)}. \tag{6.5}$$

In physical terms, the function $F$ can be considered as the “free energy function,” since it is a sum of two competing extensive terms, the entropy and an “energy” term,

$$N\alpha \log_2 (1 - \frac{1 - (1-d)^k}{2^k - 1}).$$

In the limit as $N \to \infty$ the distribution of $d$ is peaked at the value of $d$ that maximizes $F$. This maximum (or “saddle point”) is determined by the solution of the equation

$$\frac{\partial F(\alpha, d, k)}{\partial d} = 0.$$

Denoting this saddle point value of $d$ by $\tilde{d}$, we obtain

$$\log \frac{\tilde{d}}{1 - \tilde{d}} + \alpha \frac{k(1-\tilde{d})^{k-1}}{2^k + (1-\tilde{d})^k - 2} = 0 \ , \tag{6.6}$$

or equivalently

$$\alpha = \frac{2^k + (1-\tilde{d})^k - 2}{k(1-\tilde{d})^{k-1}} \log \frac{1 - \tilde{d}}{\tilde{d}} \ . \tag{6.7}$$

In this case, the error of the "Bayes algorithm," which predicts a label according to the predictions of the majority of the hypotheses in the version space, as well as the expected error of the "Gibbs algorithm," which predicts a label according to a label of a single hypothesis drawn at random from the version space, would be the error of a hypothesis with a distance $\tilde{d}$ from the target hypothesis, i.e.,

$$P(\text{error}|d) = (1 - (1 - d)^k)(1 - \gamma_1). \tag{6.8}$$

Fig. 6.1 shows the error as a function of $\alpha$, as derived from equations 6.7 and 6.8.

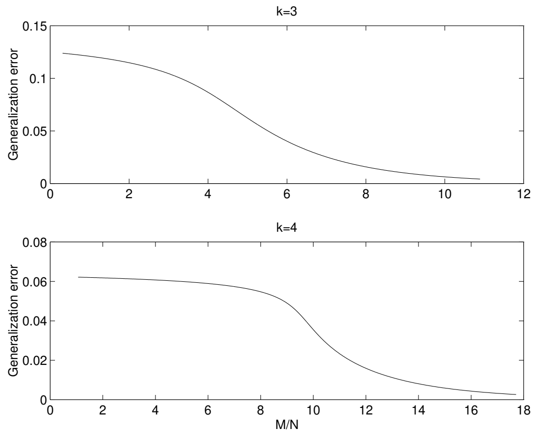


Figure 6.1: Generalization error as a function of $\alpha = M/N$ for $k = 3$ (upper figure) and for $k = 4$ (lower figure)

In order to find a hypothesis that is consistent with the sample, one needs to find a satisfying assignment to a k-CNF formula – a problem that is known to be NP-complete. The next section introduces a novel probabilistic algorithm that solves this problem.

## 6.4 The "phase-space" annealing method

### 6.4.1 Introduction

As mentioned above, stating that a certain problem is NP-Complete does not mean that an exponential number of computational steps is required for the resolution of a certain instance of the problem. Indeed, many instances of NP-Complete problems can be solved in polynomial time using various heuristics. Since many NP-Complete problems are of tremendous practical importance, many methods for solving various NP-Complete problems were developed with a considerable degree of success.

Many of the NP-Complete problems that are encountered in practical situations are optimization problems, where a certain *cost function* of many variables is needed to be maximized or minimized, possibly under certain constraints. Some basic strategies for solving such problems are "divide and conquer", constraints propagation, depth-first-search, iterative improvement, and stochastic optimization.

"Divide and conquer," as the name implies, is based on dividing the problem into subproblems of manageable size, and then solving the subproblems. The solutions to the subproblems must then be patched together. If the subproblems are naturally disjoint, and an appropriate division is performed, this method can produce very good results.

In constraints propagation, one assumes a certain value for one of the variables. This reduces the degrees of freedom in the problem, and thereby may force certain values for other variables. In this case, one assumes these values for these variables. If no variable is perfectly constrained, one proceeds to make an intelligent guess regarding the value of another variable. This process is repeated until a solution is found or until a "dead end" is reached. In the latter case, one performs "back-tracking" by undoing several steps, and the process is re-started. In the context of k-CNF, such procedures are known as Davis-Putman (DP) methods[37]. In this case, each time a given variable is assigned a certain value, all the clauses in which it participates are either eliminated (if the variable satisfies the clause) or else the size of the clause is reduced by one. A variant of this procedure, the tableau method, provides a most effective way to solve k-CNF problems for $k > 2$.

The depth-first-search strategy is a searching method that is usually performed over a tree, where the leaves of the tree are completely defined configurations, while the nodes define subsets of the configuration space. If the search is performed in the space of assignments of Boolean variables, then the nodes of depth $d$ correspond to states in which $d$ variables are fixed. At each step of the search, the most promising search direction is chosen. The process is repeated until a solution is found or until a "dead end" is reached. In the latter case, one performs "back-tracking" by undoing several steps, and the process is re-started. Thus, constraint propagation can be considered as a depth-first-search method.

In iterative improvement, one starts with a certain configuration of the variables. One then performs a search in the configuration space for a configuration that is better than the current one. If such a configuration is found, then it becomes the current configuration. This process is repeated as long as better configurations can be found by the search algorithm. Such methods are promised to reach local minima, which may be good enough for all practical purposes. There is also a chance to reach global minima, which may be enhanced if the process is repeated many times, each time using a different starting configuration.

Stochastic optimization attempts to deal with the local minima problem by allowing, with some probability, that a configuration that is worse than the current configuration will be selected. In Metropolis-type algorithms, which were first introduced for the simulation of physical systems, an energy-function $E$ is defined, and it is supposed to reflect how "bad" the configuration is. In constraint satisfaction problem, the energy of a configuration is usually the number of unsatisfied constraints. The algorithm starts with a random configuration, and at each step it searches for a better configuration (with lower "energy"). If such a configuration is found, it becomes the current configuration. If a configuration with a higher energy $E + \Delta E,\ \Delta E > 0$ is found, then with probability proportional to $e^{-\beta \Delta E}$ the new configuration is selected. The factor $\beta$ is the "inverse temperature" factor, discussed in section

1.3.4. If the "temperature" is very high, i.e., $\beta$ is very small, new configurations are rarely rejected, and the search algorithm is not very sensitive to the energy of a given configuration (in physical terms, the system is "melted"). If, on the other hand, the temperature is very low, only configurations with equal or lower energy can be selected at each step, and only local minima can be guaranteed to be obtained. The factor $\beta$ should therefore be carefully adjusted in order to get good results.

The celebrated simulated annealing algorithm, [87] allows for adjusting the value of the parameter $\beta$ during the search process. This model is inspired by the method of slow cooling, known as annealing, which is used to obtain a crystal-like structure in various materials. Using this method, the value of $\beta$ is slowly increased during the iteration of a Metropolis-like algorithm. This method was proved to be very successful in solving various optimization problems, such as the "traveling sellsman" and the design of computer chips.

In this section an alternative approach is suggested: the original problem is mapped to an easy "pseudo problem" in such a way that to each assignment in the original problem there corresponds a set of "pseudo-solutions" solving the pseudo-problem. The cardinality of this set is exponential in the number of constraints that the original assignment satisfies. A stochastic algorithm is thereafter invoked to solve the pseudo problem. The main difference between this algorithm and the traditional Metropolis and simulated annealing methods is the method by which preference to "better" configurations is induced: instead of controlling the probabilities of getting a better configuration using the exponential factor $e^{-\beta\Delta E}$, the problem is constructed such that the "phase space" of better configurations is much larger. We therefore name this method "phase-space annealing."

If:

1. most of the pseudo-solutions correspond to the assignments that satisfy the maximal number of constrains,

2. the probability to reach most of the pseudo-solutions is about equal,

then, with high probability, a satisfying assignment to the original problem can be found. It turns out that, whereas finding a construction that would fulfill the first requirement is easy, satisfying the second condition is hard in general. However, this scheme is still useful as a heuristic. In the next section we describe an algorithm for solving a 3-CNF formula, based on the above construction. The algorithm yields good results solving "hard," large 3-CNF formulae.

### 6.4.2 An algorithm for solving 3-CNF formula

We start by considering a very simple embedding of the original 3-CNF problem in a large but "easy" problem, using the following setting: Given a 3-CNF formula of $m$ clauses $C_1, \ldots, C_m$ over $n$ variables, $(x_1, \ldots, x_n) \in \{0,1\}^n$:

$$f(\mathbf{x}) = (\bar{x_a} \vee x_b \vee x_c) \wedge (x_d \vee x_e \vee \bar{x_f}) \ldots \tag{6.9}$$

A pseudo-formula $g(\mathbf{x},\mathbf{y})$ is constructed, by adding $n^2$ "dummy variables", in the following manner:

$$\begin{aligned} g(\mathbf{x},\mathbf{y}) = &(\bar{x_a} \vee x_b \vee x_c \vee y_1^1) \wedge (\bar{x_a} \vee x_b \vee x_c \vee y_2^1) \ldots (\bar{x_a} \vee x_b \vee x_c \vee y_n^1) \\ &\wedge (x_d \vee x_e \vee \bar{x_f} \vee y_1^2) \wedge (x_d \vee x_e \vee \bar{x_f} \vee y_2^2) \ldots (x_d \vee x_e \vee \bar{x_f} \vee y_n^2) \ldots \end{aligned} \tag{6.10}$$

where $y_i^j$ is a Boolean variable, $y_i^j \in \{0,1\}$. Since each $y$ variable appears in only one clause, finding a satisfying assignment to the formula 6.10 is very easy, (e.g., by setting the value of all the $y$'s to "1".) Furthermore, any satisfying assignment to the original formula $f(x)$ yields the $n^2$ $y$ variables free, and therefore has a degeneracy of $2^{n^2}$.

Starting from a random assignment, the algorithm attempts to find a satisfying assignment for $g(\mathbf{x},\mathbf{y})$ by picking at random an un-satisfied clause and randomly inverting a variable in the clause. Observe that inverting a variable in an un-satisfied clause necessarily causes that clause to be satisfied [2]. This procedure is repeated until a satisfying assignment for $g(\mathbf{x},\mathbf{y})$, $(\mathbf{x}',\mathbf{y}')$, is found. The algorithm proceeds to check if the original formula $f(\mathbf{x})$ is satisfied by $\mathbf{x}$. If it does, the problem is solved. If it does not, then the procedure is repeated, starting with a new random configurations, until a pre-defined "futility index" is reached, in which case the problem is declared "probably unsatisfiable". This simple algorithm works well for moderately large random formula (less than 100 variables).

### 6.4.3 Improvements to the basic algorithm

In order to enhance the efficiency of the algorithm, two basic improvements were added:

1. Instead of replicating each clause $n$ times, we attach to each clause $C_i$ a counter: $c_i = \sum_{j=1}^{n_y} y_j^i$, which counts the number of "y" variables, associated with the clause $C_i$, that have a value "1". The counters are reduced at times by one, using a procedure that mimics the way in which this number is reduced in the basic setting, but the counters are not allowed to vanish[3]

2. A simulated-annealing-like procedure is also incorporated in the algorithm, allowing changes in the x's values that increase the number of unsatisfied clauses with some probability.

The resulted algorithm is as follows:

**Algorithm Phase-space annealing**

**Input:**

- A k-CNF formula with $n$ variables and $m$ clauses.

[2] This heuristic, first introduced by Papadimitriou[130], is also referred as "the random walk heuristic"

[3] This setting has some similarity to methods that associate weights to each clause, and reduce the weights during the iterations[160].

- Maximal number of iterations.
- Three parameters, $\beta_{min}$, $\beta_{max}$ and $\gamma > 1$ that control the "annealing process".

**Initialize** the $m$ counters of the clauses: $c_i = \gamma \cdot n$.

**Start with a random assignment** $A$

**Do** while there are unsatisfied clauses or while number of iterations < maximal number of iterations:

1. Define $\mathrm{SAT}_i(A) = 0$ if the assignment $A$ satisfy $C_i$ and $\mathrm{SAT}_i(A) = 1$ otherwise.
2. Evaluate $E = \sum_i c_i \mathrm{SAT}_i(A)$.
3. Pick at random clause $C_i$ that is not satisfied by the current assignment, according to a probability:
$$\mathbf{P}(i) = c_i/E.$$
4. With probability $p$ reduce $c_i$ by 1 (this simulate a situation in which a "y-variable" is flipped)
5. with probability$1 - p$:
   (a) Pick at random one variable that appear in $C_i$ and flip its value.
   (b) Evaluate the new value of $E$ and the difference $\Delta E$ between the current value of $E$ and the previous value.
   (c) Set $\beta = \beta_{max} - \frac{E}{\gamma M N}(\beta_{max} - \beta_{min})$ (i.e., the system is "cooling down" as the energy decreases.)
   (d) If $\Delta E > 0$ Then, with probability $e^{-\beta \Delta E}$ flip back the spin.

**Output** $A$

For reasons that will become clearer in the next chapter, a typical 3-CNF formula is actually hard only when the ratio between the number of clauses $M$ and the number of variables $N$, $\alpha = M/N$, is close to $\alpha_c \approx 4.27$.

The following table shows the median number of spin-flips which was required for the resolution of satisfiable 3-CNF formulae. (The median was evaluated using a sample of 1000 formulae)

| # of variables | # of clauses | # of flips |
|---|---|---|
| N=100 | 427 | 4988 |
| N=200 | 854 | 27952 |
| N=300 | 1281 | 98000 |

(6.11)

### 6.4.4 Comparison with other methods

There are two general approach for the resolution of k-CNF formulae: Depth-first search, or constraint propagation methods, which are variants of the Davis-Putman (DP)[37] methods, and local search, or "energy minimization" methods, such as the method described above. The main advantage of constraint propagation methods is their ability to prove that a formula is not satisfiable by showing inconsistencies. This is in contrast with the local search methods, which are only able to found existing solutions. However, constraint propagation methods are, in practice, limited to problems with several hundreds of variables at the hard regime, while local search methods have no such limitation.

The best variant of the Davis-Putman method is the Tableau heuristic of Crawford and Auton[35]. Since the basic iterative steps in local search ("spin-flip") and in DP methods ("DP call") are rather different, the comparison with Tableau method was based on average run-time on similar machines[4] For N=300 and M=1281, the average run time of the "phase-space annealing" algorithm was 22 sec. (without a special attempt for optimization) on an SGI 175 MHz. machine, while the reported average run time of the Tableau algorithm was 141 sec on Sparc 10.51[35].

The best published local search methods for solving k-CNF formulae is, most probably, the "biased random-walk" method of Selman & Kautz[160][161], which is based on the following heuristic: on each step, with probability $p$ it select an unsatisfied clause at random and flip a variable inside it, and with probability $1-p$ it performs a "greedy move": i.e., it flip a variable that yields the greatest number of satisfied clause (either from the variables in the chosen clause or, in another variant, from all the variables in the problem.) Since no results regarding the performance of the algorithm for $n > 200$ were available, the two basic variants of the "biased random walk" algorithm were programmed. The results were found to be considerably inferior to the performance of the "phase-space annealing" methods, and for $n > 150$ only small fraction of the solution were found in less then $10^6$ iterations. This, however, still does not prove that the "biased random walk" is indeed inferior, since there are a lot of "tricks of the trade" that are involved in the implementation of heuristics. We intend to check the performance of the algorithm in the next challenge competition.

## 6.5 Summary

In this chapter we discussed the relation between supervised learning and constraint satisfaction problems, and analyzes the dual problem of CNF learning as a prototype. The average sample complexity of this problem was analyzed using a method inspired from statistical physics. A a novel learning algorithm, based on a new "phase-space annealing" method was introduced. This algorithm achieves good performance in solving typical k-CNF formulae, a problem of tremendous practical importance. The implementation of the "phase space annealing" paradigm in this setting was rather naive, and one can think of more sophisticated manner to implement the basic idea. (e.g., by allow dummy variables to appear in more

[4] According to the Matlab standard benchmark

then one clause.) Still, even this naive implementation yield surprisingly good results solving large k-CNF formulae in the hardest regime. We hope that this method can serve as a new paradigm that will inspire further research in this direction.

# Chapter 7

# Characteristics of typical k–SAT problem and the average computational complexity

## 7.1 Introduction

We study several typical-case characteristics of K-SAT by considering an ensemble of randomly generating k-SAT formulae. For each formula, we generate $M$ clauses, where each clause is generated by randomly selecting $k$ variables from the set of $N$ variables and negating each selected variable with probably 0.5 to construct the clause. Formulae constructed at random, keeping the ratio of clauses to variables, $\alpha = M/N$ , constant as $M, N \to \infty$, provide a natural ensemble of test problems.

1-SAT formula is simply a conjunction of $M$ literals. Once a variable and its negation appear in the formula, the formula becomes unsatisfiable. Denote by $m_+$ the number of variables that are not negated and by $m_- = M - m_+$ the number of negated variables. The probability that a random formula is satisfiable is the ratio between the number of satisfiable formulae and the total number of formulae:

$$\frac{\sum_{m_+=0}^{M} \binom{N}{m_+}\binom{N}{M-m_+}}{\binom{2N}{M}}. \tag{7.1}$$

and for $M = O(\sqrt{N})$ the problem becomes unsatisfiable. 1-SAT can easily be solved in linear time.

2-SAT formula is a conjunction of $M$ clauses, each of which is a disjunction of two literals. 2-SAT can be solved efficiently, i.e., in time polynomial in the size of the formula (in fact, a linear time algorithm is known [30]) and it therefore belongs to the class “P” (polynomial time solvable problems). If the number of clauses $M$ is small, relatively to the number of variables $N$ (i.e., small $\alpha$), we expect that the problem would be under-constrained, and therefore most of the formulae would be satisfiable. At high values of $\alpha$, the problem is typically over-constrained and most of the formulae are unsatisfiable. An upper bound for a value of $\alpha$ beyond which most formulae would be unsatisfiable can be found using this simple argument: since each clause by itself is satisfied by 3/4 of the assignments (provided that no variable appears twice in a clause), the average number of satisfying assignments, over the entire ensemble of formulae with $M$ clauses and $N$ variables, is

$$(3/4)^M 2^N = 2^{N\cdot(\alpha \log(3/4)+1)} \tag{7.2}$$

This number goes exponentially to zero for $\alpha > -1/\log(3/4) \approx 2.409$, and therefore almost all the formulae have to be unsatisfiable beyond that value[31] [40]. There are, however, two remarkable facts regarding the transition from the SAT to the UN-SAT phase, i.e., from the range in which most formulae are satisfiable to the range in which most formulae are not:

- The transition happens at a critical value $\alpha_c = 1$, i.e. when the number of clauses is equal to the number of variables[30].
- The transition is very abrupt: by adding a small number of constraints, most of the formulae become unsatisfiable, despite the fact that each formula has an entirely different set of constraints. We refer to this phenomenon as the existence of a "threshold" for $\alpha$.

Although by now the nature of the transition for the 2-SAT problem is well understood, the explanation is far from being simple. We discuss these issues latter.

For $k \geq 3$, k-SAT is NP-Complete. Computer experiments suggest the existence of thresholds for $k = 3, 4, 5,$ and 6[88]. Recently, it has been shown rigorously that a threshold exists for any value of $k$[56], but determining the exact location of the threshold remains a difficult open problem.

The threshold phenomenon of the $k - SAT$ problem attracts considerable attention from scientists in the fields of theoretical computer science, statistical physics, and combinatorics (e.g., [19],[1][29][30][35][40] [41][51][82][86][88][90][109][110] [111][112][113][56]). Among the questions addressed are:

- What is the functional shape of the threshold?
- Is there a relation between the nature of the threshold and the computational complexity of a typical problem?
- How can the exact value of the threshold be located?

In the following sections these questions and several others will be addressed.

## 7.2 The shape of the threshold

For any finite $N$, the interval of $\alpha$ over which most of the formulae become unsatisfiable is finite. The functional form of the threshold, $f(k, \alpha, N)$ is the probability that a formula, drawn at random from an ensemble of formulae with $N$ variables and $M = \alpha N$ clauses with $k$ variables each, is unsatisfiable, for $\alpha$ close to $\alpha_c$.

This functional form was first studied experimentally by Scott Kirkpatrick and Bart Selman[86, 88], where interesting scaling behavior was found: for $k > 2$ a scaling exponent $\mu(k)$ was found, such that $f(k, \alpha, N^{1/\mu(k)})$ is almost independent of $N$. Furthermore: if $\alpha$ is replaced by a re-scaled variable

$$\tilde{\alpha}(k) = \frac{\alpha - \alpha_c(k)}{\alpha_c(k)},$$

then the curves of $f(k, \tilde{\alpha}(k), N^{1/\mu(k)})$ for $k > 2$ are very close to each other (cf. fig. 7.1) Such scaling behavior is known from the theory of disordered materials, but at first there was no theoretical understanding of the functional form of curves and the scaling exponents.

In order to gain a better understanding of this phenomenon, I have tried to analyze the threshold using purely probabilistic arguments: consider the ensemble of all k-CNF formulae with $M$ clauses and $N$ variables, for some fixed $k$, $M$, and $N$. Denote by $T$ the set of all the formulae in the ensemble and its cardinality by $|T|$ and by $T_i$ the set of all formulae that are satisfied by a given assignment $\mathbf{x}_i$ and its cardinality by $|T_i|$. One has:

$$\frac{|T_i|}{|T|} = \left(1 - 2^{-k}\right)^M = \gamma_0^M. \tag{7.3}$$

independently of $i$. (This is due to the "gauge symmetry" of the ensemble, i.e., to the fact that if some of the variables are negated in all assignments and in all formulae in the ensemble, the entire ensemble remains the same.)

The probability $f(k, \alpha, N)$ can be obtained, in principle, using the inclusion-exclusion formula,

$$\begin{aligned} P_{SAT} = & \frac{1}{|T|} \sum_i |T_i| - \frac{1}{|T|^2} \sum_{i \geq j} |T_i \cap T_j| \\ & + \ldots \frac{1}{|T|^{2^N}} |T_1 \cap T_2 \cap \ldots T_{2^N}|) \ . \end{aligned} \tag{7.4}$$

Evaluating this sum is generally impossible for large $N$. In fact, even the first terms are quite complicated. However, if it is the case that

$$\frac{1}{|T|^n} |T_1 \cap T_2 \cap \ldots T_j| = \left(\frac{|T_i|}{|T|}\right)^j,$$

(say, if the Hamming distance between all the assignments is exactly $\frac{1}{2}$), all the elements in the $j$-th term in the sum are equal to $(\gamma_0^M)^j$. In this limiting case the sum can be rewritten as,

$$P_{SAT} = \sum_{j=1}^{2^N} \binom{2^N}{j} (\gamma_0^M)^j = 1 - \left(1 - \gamma_0^M\right)^{2^N} \ , \tag{7.5}$$

and thus,

$$P_{UN-SAT} = \left(1 - (\gamma_0^M)\right)^{2^N} = \left(1 - (\gamma_0^\alpha)^N\right)^{2^N} \quad . \tag{7.6}$$

Using the re-scaled parameter

$$\tilde{\alpha} \equiv \frac{\alpha - \alpha_c}{\alpha_c}$$

($\alpha = \alpha_c(\tilde{\alpha}+1)$.) Equation 7.6 can now be rewritten as

$$P_{UN-SAT} = \left(1 - (\gamma_0^{\alpha_c})^{(\tilde{\alpha}+1)N}\right)^{2^N} \quad . \tag{7.7}$$

In the above approximation

$$\alpha_c = \frac{\ln\frac{1}{2}}{\ln(\gamma_0)}, \Longrightarrow \gamma_0^{\alpha_c} = \frac{1}{2}$$

and

$$P_{UN-SAT} = \left(1 - \frac{1}{2}^{(\tilde{\alpha}+1)N}\right)^{2^N}$$
$$= \left(1 - \frac{2^{-\tilde{\alpha}N}}{2^N}\right)^{2^N} \longrightarrow e^{-2^{\tilde{\alpha}N}} \quad . \tag{7.8}$$

We refer to this limit as the "annealed limit".

This functional limit form serves as a good approximation for $k > 2$, providing that[86, 88]

- The approximate critical $\alpha$, $\alpha_c = \ln\frac{1}{2}/\ln(\gamma_0)$ is replaced with the correct critical $\alpha$ (which, for $k > 2$ can only be obtained from numerical simulations).
- The number of variables is scaled, $N \longmapsto N^{\frac{1}{\nu}}$, where the exponent $\nu = \nu(k)$ is needed to be found numerically.

Figure 7.1 shows the convergence with $k$ of the scaled approximate formula for $k$ =2,3,4,5 and 6 to our analytic limiting function. As can be seen there is a very good agreement with the re-scaled experimental curve using the form in equation 7.8 for $k > 2$, which improves as $k$ increases.

## 7.3 Computational complexity and thresholds

It is commonly believes that $P \neq NP$, and that, for every NP-Complete problem (e.g., 3-CNF), the resolution of some instances of the problem requires an exponential number of steps. This does not say much regarding the typical case complexity of instances of the family.

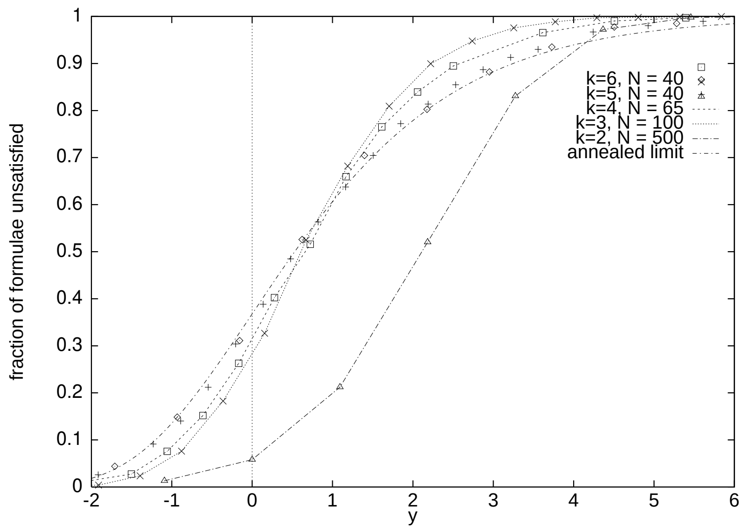


Figure 7.1: Scaled probability of UN-SAT near the transition.

Indeed, for many NP-Complete problems, heuristic algorithms that solve many instances of the problem in polynomial time do exist. It is therefore of great practical interest to characterize the *average-case complexity* of NP-Complete problems. However, this question did not receive the attention it deserves in theoretical computer science and very little is known regarding the average-case complexity. This state of affairs is probably due to the fact that estimating the average-case complexity is a hard task. On the other hand, proving that a problem is NP-Complete is relatively straight-forward: all that is required is to show a reduction of one of the several thousands of known NP-Complete problems to the problem at hand.

The threshold phenomenon sheds new light on the characterization of average-case complexity: NP-Complete problem are usually stated as decision problems. In the 3-CNF SAT problem, one needs to decide whether a certain 3-CNF formula is satisfiable. Since well below the threshold almost all of the formulae are satisfiable, and well above the threshold almost all of the formulae are unsatisfiable, the decision in these two regions should be easy. Indeed, computer experiments show that: a) well below the threshold heuristic algorithms can find, in most cases, a satisfying assignment in polynomial time, b) well above the threshold most formulae can be proved to be unsatisfiable in polynomial time. The real "hard core" of typical-case complexity lies therefore in the immediate vicinity of the threshold, where all experiments show an exponential scaling of the number of computational steps required for the solution.

## 7.4 Moments of satisfaction: the statistical mechanics of k-CNF satisfaction

Threshold phenomena and phase transitions are well known in statistical physics. Statistical physics is a natural framework for studying typical behavior of systems with large numbers of degrees of freedom. In particular, the theory of spin glasses is particularly useful for the study of NP-Complete problems: one of the most useful heuristics for solving NP-Complete problems, the simulated annealing algorithm[87], discussed in section 6.4, is based on analogies from spin-glass theory, and deep connections between NP-complete problems and models studied in statistical physics were established by Y. Fu and P. W. Anderson[58]. Studying the phase transition in k-CNF using a spin-glass model and replica theory was first suggested by N. Tishby[171]. A statistical physics model for the problem can be constructed as follows[86]: since the N inputs are binary, each assignment can be represented as a vector **S** of Ising spins, $S_i \in \{-1, +1\}, i = 1, \ldots, N$. The cost function of an assignment **S** is given by:

$$E[\Delta, S] = \sum_{\ell=1}^{M} \delta\left[\sum_{i=1}^{N} \Delta_{\ell,i}\, S_i; -k\right] \tag{7.9}$$

where $\delta[i;j]$ denotes the Kronecker symbol, and

$$\Delta^{jl} = \begin{cases} 0 & \text{if } s_l \text{ does not appear in } C_j \\ 1 & \text{if } s_l \text{ appear un-negated in } C_j \\ -1 & \text{if } s_l \text{ appear negated in } C_j. \end{cases} \tag{7.10}$$

characterize a formula.

The cost function 7.9 evaluates the number of unsatisfied clauses by the assignment **S**, and can be naturally referred as the *energy function* associated with an assignment (or "configuration") **S**. The energy is "0" if the assignment satisfies the formula and is greater than zero otherwise.

One can then construct a partition function associated with a formula $\mathcal{F}$

$$Z_{\mathcal{F}} = tr_{\{s_i\}}\exp(-\beta E_{\mathcal{F}}), \tag{7.11}$$

where $\beta$ is the inverse of a fictitious "temperature" (i.e., a measure of tolerance with regard to unsatisfied clauses). If $\beta \to \infty$, i.e., the temperature goes to zero, then $Z$ gets positive contributions only from assignments that satisfy the formula, and the value of $Z(\mathcal{F})$ is the number of the satisfying assignments.

The *entropy* of the formula $\mathcal{F}$ is defined as $\mathcal{S}(\mathcal{F}) = \log(Z(\mathcal{F}))$. Again, when the temperature is zero, the entropy is the logarithm of the number of satisfying assignments.

Above the threshold, most of the formulae do not have any satisfying assignments. However, an exponentially small number of formulae may still have an exponentially large number of satisfying assignments. The average number of satisfying assignments with respect to the entire ensemble of formulae may therefore be finite at the threshold. Indeed, for all the values of $k$ for which the value of the threshold is known, the average number of satisfying assignments, $(1-2^{-k})^{\alpha_c N} 2^N$, is exponentially large at the threshold.

The average of the entropy over the entire space of formulae,

$$\bar{\mathcal{S}} = \langle \log(Z(\mathcal{F})) \rangle_{\mathcal{F}}, \tag{7.12}$$

should be a better predictor for the threshold, since it is much less sensitive to the small fraction of formulae with a large number of satisfying assignments. Furthermore, in many cases quantities of this form have the "self-averaging" property, which means that the average over the entire ensemble has the same basic properties as a typical instance. However, evaluating such quantities for disordered systems is very complicated. One method that was found successful in many cases is the "replica method"[106]. In a nutshell, this method is based on the observation that for a real variable $z$, the function $\log(z)$ can be written as :

$$\log(z) = \lim_{n \to 0} \frac{z^n - 1}{n} \tag{7.13}$$

and therefore one may attempt to consider a heuristic for solving the expression for the average entropy $\bar{\mathcal{S}}$, using:

$$\bar{\mathcal{S}} = \langle \log(Z(\mathcal{F})) \rangle_{\mathcal{F}} = \left\langle \lim_{n \to 0} \frac{Z^n - 1}{n} \right\rangle \tag{7.14}$$

and taking the average over $Z^n$ (i.e., the **moment** of order $n$)[1] by considering average over the product of "replicae" that are identical copies of the disordered system (in our case – random formulae).

A natural order parameter for disordered systems is the Edwards-Anderson parameter, which is the average overlap between the average values of the variables in any two replicae. Under the self-averaging assumption, this parameter can be evaluated using the same method that was used for the derivation of equation 6.7. Indeed, the theoretical curve from equation 6.7 perfectly agrees with the values of the Edwards-Anderson parameter that was obtained in numerical simulations [86].

If the structure of the space of the satisfying assignment undergoes a substantial change at the threshold, one would expect that this change would be reflected in some way in the curve $q(\alpha)$. Indeed, the maximal change in the value of the slope of $q(\alpha)$ happens at $\alpha = 4.22$, a value that is closer to the experimental value of the threshold (4.17-4.27) than any other theoretical result known to date. For $k = 4$ the maximum in the slope of $q(\alpha)$ is obtained at $\alpha = 9.58$, which is also consistent with the known experimental value $9.7 \pm 0.15$. For $k = 2$, however, the maximum in the slope of $q(\alpha)$ is obtained at $\alpha = 1.2$, while the value of the threshold is known for sure to be 1. As we shell see in section 7.5, the phase transition is indeed smoother for $k = 2$, and there is no discontinuous change in the order parameters at the threshold. This may be a possible explanation for the in-ability to predict the threshold from the average overlap between assignments.

[1]This is the origin of the name "moments of satisfaction"

For $k > 4$ the function $\alpha(\tilde{q})$ that we derived from equation (11) is no longer single-valued for all $\alpha$. This is typical for a first order transition, where two stable solutions and one unstable solution occur together between two spinodal points. In terms of the overlap, there is a range of $\alpha$ where there are two local maxima in this distribution of the overlap between satisfying assignments for a typical formula. The bi-modality of this distribution reflects the fact that the space of satisfying assignments splits into two clusters ("replica symmetry breaking").

Figure 7.2 depicts the experimental ensemble averages, over 100 random formulae, of the overlap distributions with $k = 5$, $N = 25$ and $\alpha$ between 16 and 20. As can be seen, at the value $\alpha = 18$ the distribution becomes bi-modal, where the most likely value of the overlap jumps discontinuously from a low value (near 0) to a much higher value (close to 1), as the theory predicts. Figure 7.3 gives the overlap histogram for a single (typical) formula with $k = 5$, $N = 25$ and $\alpha = 19$, to demonstrate the self-averaging phenomenon.

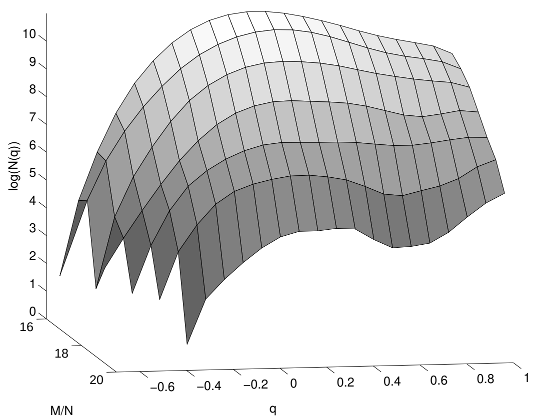


Figure 7.2: Ensemble average overlap histogram.

The functions $q(\alpha)$ and $F(\alpha(\tilde{d}))$ are plotted for $k = 5$ in figure 7.4. For $k = 5$ the effective $F$ shows a discontinuous change in its derivative at the value $\alpha = 20.6$, – very close to the numerical value of the SAT-UN-SAT transition, $\sim 20.9$. For $k = 6$ the overlap transition occurs at $\alpha = 42.8$ – again, just below the numerical value of the SAT-UN-SAT transition, $\sim 43.2$. The discontinuous jump in the overlap is due to the fact that the local maximum with the higher $\tilde{q}$ becomes the global maximum of $F$ at this value of $\alpha$. This change corresponds to a sharp change in the structure of the assignments space from typical small overlap between assignments to a much smaller set of assignments with high overlap, just before they disappear completely. This structural shift can have significant computational ramifications, particularly for the design of random algorithms.

The replica model was later developed by Monasson and Zecchina[109], [110]. In their work, they were able to explain the phase transition for $k = 2$ using replica-symmetric calculations, and to show that for $k \geq 3$ the system exhibits a replica symmetry breaking transition.

In section 7.5 we show that the threshold is characterized by the appearance of "back-

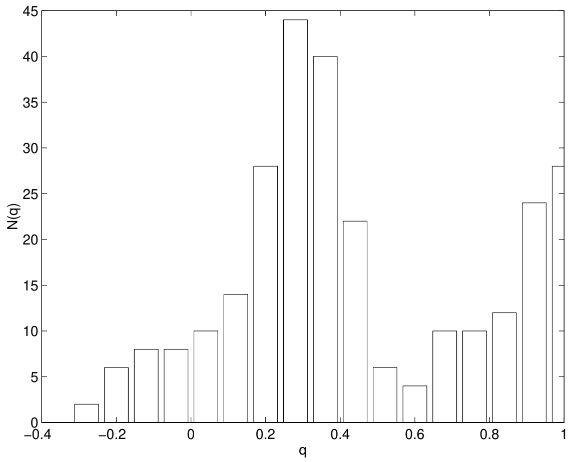


Figure 7.3: Overlap histogram for a single formula in the bimodal regime.

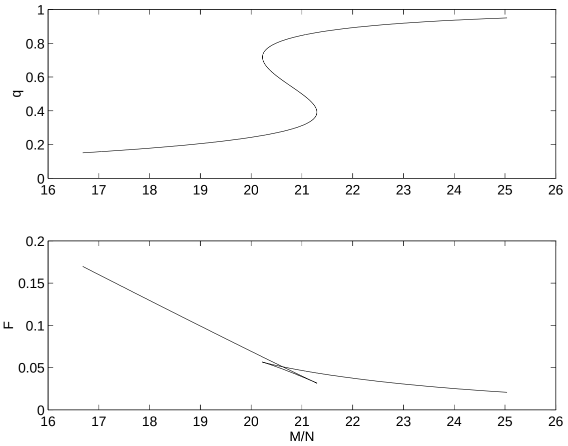


Figure 7.4: Theoretical $q$ and $F$ as functions of $\alpha$ for k=5.

bones": variables whose values are the same in all satisfying assignments. The fraction of backbones (namely, the relative number of variables whose values become fixed) is zero below the threshold and is finite above the threshold. However, it turns out that for k=2 the relative fraction of the backbones is a continuous function of $\alpha$, while for $k \geq 2$ the fraction is a discontinuous function of $\alpha$. This may explain the discrepancy in the prediction of the threshold for $k = 2$.

## 7.5 The 2+p SAT problem and the origin of computational complexity

While any k-CNF formula with $k > 1$ undergoes a phase transition at a critical value of $\alpha$, the 2-SAT problem can be solved in linear time while for $k > 2$ k-SAT problems are NP-Complete. Furthermore, the replica calculations of Monasson and Zecchina suggested that, while for $k = 2$ the value of the threshold can be derived using the replica-symmetric ansatz, no such derivation is possible for $k > 2$.

In order to gain a better understanding of this substantial difference and its implications regarding the issue of typical computational complexity, I have joined the collaboration of Monasson, Zecchina, Kirkpatrick and Selman, [111][112] [113] in the study of the model of "2+p-SAT", which smoothly interpolates between the 2-SAT and the 3-SAT problems: consider $N$ Boolean variables and a random CNF formula $F = F_2 \wedge F_3$ including $M$ clauses such that $F_2$ (respectively $F_3$) is a random 2-CNF (resp. 3-CNF) formula with $(1-p)M$ (resp. $pM$) clauses. $F_2$ and $F_3$ are drawn independently from each other and $0 \leq p \leq 1$. As a consequence, the model smoothly interpolates between 2-SAT ($p = 0$) and 3–SAT ($p = 1$). It is worth noticing that as far as the complexity classification of the problem is concerned, for any $p > 0$ any instance of the model contains a 3-CNF sub-formula, therefore proving that the problem itself belongs to the NP-complete class. Of course, NP-completeness is a worst case measure. Our interest here is in the complexity of more "typical" problem instances.

Let $\alpha_c(p)$ denote the threshold ratio $M/N$ of the above model at fixed $p$. From the known results regarding the threshold for $k = 2$ and $k = 3$, $\alpha_c(0) = 1$ and $\alpha_c(1) \simeq 4.22$. In addition, $F$ cannot be satisfied ("almost always") if the number of 2-clauses (respectively 3-clauses) exceeds $N$ (resp. $4.22N$). As a consequence, the critical ratio must be bounded by $\alpha_c(p) \leq \min\left(\frac{1}{1-p}, \frac{4.22}{p}\right)$.

### 7.5.1 Statistical mechanics theory of the 2+p-SAT problem

The above mixed random SAT problem, hereafter referred to as $2+p$-SAT, can be mapped onto a diluted spin energy-cost function 7.9, as discussed in section 7.4. In this model, we have used the following cost-energy function

$$E[\Delta, S] = \sum_{\ell=1}^{M} \delta \left[ \sum_{i=1}^{N} \Delta_{\ell,i} \, S_i ; -k \right] \tag{7.15}$$

where $\delta[i;j]$ denotes the Kronecker symbol. By imposing the constraints

$$\sum_{i=1}^{N} \Delta_{\ell,i}^2 = \begin{cases} 2 \,, & \forall \ell = 1, ..., (1-p)M \\ 3 \,, & \forall \ell = (1-p)M + 1, ..., M \end{cases} \tag{7.16}$$

we ensure that the fraction of clauses of lengths 2 and 3 are respectively equal to $(1-p)$ and $p$. The ground state (GS) properties of the cost function (7.15), i.e., its minima, encode for the existence of satisfying assignments (zero violated clauses, $E_{GS} = 0$) or, if no satisfying assignment exists, for assignment with minimum number ($E_{GS} > 0$) of violated clauses, which is the MAX-SAT problem.

The above expression for the cost-energy function can also be written in a way that is manifestly reminiscent of spin-glass models (and more precisely neural networks with an extended Hebbian rule [75])[2],

$$E[\Delta, S] = \frac{\alpha}{2^k} N + \sum_{R=1}^{k} (-1)^R \sum_{i_1 < i_2 < \ldots < i_R} J_{i_1, i_2, \ldots, i_R} S_{i_1} S_{i_2} \ldots S_{i_R} \quad , \tag{7.17}$$

where the couplings are defined by

$$J_{i_1, i_2, \ldots, i_R} = \frac{1}{2^k} \sum_{\ell=1}^{M} \Delta_{\ell, i_1} \Delta_{\ell, i_2} \ldots \Delta_{\ell, i_R} \quad . \tag{7.18}$$

Due to absence of any (site) correlations between the components of the random matrices $\Delta$, k-SAT appears to be a mean-field spin glass model and can therefore be solved using the replica approach [110]. Using this method, the calculation of the average over the disorder leads to an effective energy or cost function which consists of many identical copies of one instance of the model system with the dynamical variables (usually Ising spins) in different replicas coupled by some non-linear function. The most striking feature of this approach is the natural emergence, in the large $N, M$ limit and at fixed $p$ and $\alpha$, of some order parameters describing the statistical structure of Boolean assignments satisfying all clauses. To exemplify the meaning of such an order parameter, assume that $\alpha < \alpha_c$ and call $\mathcal{S}^g = (S_1^g, S_2^g, \ldots, S_N^g)$, $g = 1, \ldots, \mathcal{N}_{GS}$ the $\mathcal{N}_{GS}$ ground states (with minimum energy). Define $m_i$ to be the average value of spin $S_i^g$ over all solutions: $m_i = \sum_g S_i^g / \mathcal{N}_{GS}$. Clearly, $m_i$ ranges from $-1$ to $+1$ and $m_i = -1$ (respectively $+1$) means that the corresponding Boolean variable $x_i$ is always false

[2]One may notice that the above model consists of a mixture of a Sherrington-Kirkpatrick diluted model ($k = 2$) and the so-called p-spins or Potts model ($k = 3$), which are known to have rather different thermodynamical behaviors.

(resp. true) in all solutions $\mathcal{S}^g$. The quantities $m_i$'s therefore measure the variabilities of variables $x_i$ among the set of solutions, e.g., $m_i = 0$ if $x_i$ is true for half of the solutions and false in the remaining ones. The histogram $P(m)$ of all $m_i$'s gives access to some information on the microscopic correlations between solutions. In other words, the accumulation of Boolean magnetizations $m$ around $\pm 1$ account for almost completely constrained variables, whose logical values cannot vary from solution to solution, while the center of the distribution $P(m \simeq 0)$ signals weakly constrained variables.

As previously mentioned, Due to the mean-field nature of k-SAT, thermodynamical properties are naturally derived as optima of functional expressions over a set of functional order parameters. It turns out that, for $\alpha < \alpha_c$, the onset of ordering can be identified by the fact that a single stable state occurs in multiple replicas which corresponds to a physical state that is highly irregular in structure due to the randomness of the problem. This symmetry property, is usually referred as Replica Symmetry (RS). [3] Using the RS scheme, the order parameter is precisely the magnetization distribution $P(m)$ defined above. The optimization condition reads as a self-consistent equation for $P(m)$,

$$\begin{aligned} P(m) = \quad & \frac{1}{1-m^2} \int_{-\infty}^{\infty} du \ \cos\left[\frac{u}{2}\ln\left(\frac{1+m}{1-m}\right)\right] \times \\ & \exp\left[-\alpha(2+p) + 2\alpha(1-p)\int_{-1}^{1} dm_1 P(m_1)\cos\left(\frac{u}{2}\ln\left(\frac{1-m_1}{2}\right)\right) + \right. \\ & \left. 3\alpha p \int_{-1}^{1} dm_1 dm_2 P(m_1)P(m_2)\cos\left(\frac{u}{2}\ln\left(\frac{3-m_1-m_2-m_1 m_2}{4}\right)\right)\right] \end{aligned} \tag{7.19}$$

By solving the above equation, one obtains $P(m)$ and the ground state entropy. According to the meaning of the magnetizations distribution exposed above, the threshold $\alpha_c$ will coincide with the appearance of an extensive number of fully constrained variables $x_i$, i.e., an accumulation of a finite probability weight in $m = \pm 1$. This microscopic criterion, which is accompanied by a non zero GS macroscopic cost-energy, allows to localize the SAT/UNSAT critical threshold.

From the theory, there appears an essential qualitative difference between 2-SAT and 3-SAT in the way the order parameter $P(m)$ changes at the threshold. This discrepancy can be seen on the fraction $f_k(\alpha)$ of Boolean variables that become fully constrained at and above the threshold. As said above, $f_k(\alpha)$ is identically null below the threshold. For 2-SAT, $f_2(\alpha)$ becomes strictly positive above $\alpha_c = 1$ and is continuous at the transition: $f_2(1^-) = f_2(1^+) = 0$. By contrast, $f_3(\alpha)$ displays a discontinuous change at the threshold: $f_3(\alpha_c^-) = 0$ and $f_3(\alpha_c^+) = f_c > 0$ This confirms some numerical investigations showing that a rather large fraction of the Boolean variables have the same value (either always true or false) in all solutions at the threshold [41].

[3] A more subtle kind of ordering (called Replica symmetry-breaking) occurs when distinct stable states of this sort are found in different subsets of the replicas. The nature of this new type of transition is not well understood because (a) the physical interpretation of replicas is uncertain, and (b) spin glass models with realistic connectivity are difficult to explore experimentally, either on real substances or by computer simulation.

For the mixed $2+p$-SAT model, the key issue is therefore to understand how a discontinuous 3-SAT-like transition mechanism may appear when increasing $p$ from zero to one and to see how much it affects the computational cost of finding solutions at the threshold $\alpha_c(p)$. Very schematically, the analytical results we have derived by applying the method of ref.[110], are the following. (see Fig. 1).

i) for $p < p_0$ ($p_0 = 0.413$), there is a **continuous** SAT/UNSAT transition at

$$\alpha_c(p) = \frac{1}{1-p} \quad . \tag{7.20}$$

The RS theory appears to be correct up to $\alpha_c(p)$, thus identifying both the critical ratio and the typical number of solutions, as in the well known case of $k = 2$. For $p < p_0$, the model shares the same physical features as the random 2-SAT.

ii) for $p > p_0$, the RS transition becomes **discontinuous** and leads to an upper bound for the threshold (as happens in the 3–SAT analysis). For $p > p_0$, the random 2+p-SAT problem shares the same physical features as the random 3-SAT.

The typical entropy, i.e., the logarithm (base 2) of the typical number of existing solutions, can be computed exactly within the RS scheme for any $p$ and $\alpha < \alpha_c(p)$. The entropy at the transition point decreases as a function of $p$, from 0.55 for $p = 0$ to 0.13 for $p = 1$, with a change of slope around $p_0$.

It should be pointed out that the overall results described above are based on the assumption that the RS theory correctly describes the SAT phase and the qualitative nature of the transition. This is consistent with the current knowledge in the field of spin glass theory. A complete theory, also capable of describing the UNSAT phase, is under study.

In the following section, we report numerical simulations that corroborate the above results, both concerning the threshold values below $p_0$ and the change of the nature of the transition at $p_0$.

### 7.5.2 Experimental Tests

We have run experiments to test the theory between $k = 2$ and $k = 3$, finding thresholds and assigning an exponent $\nu$ for narrowing the critical region by finite-size scaling. We find (Fig. 1) good agreement between the observed and predicted values of $\alpha_c$, with an error that increases slowly from $p = 0.413$ to $p = 1.0$. We also show the bounds obtained by rigorous methods in Fig. 1. Lower bounds are obtained by showing that some analyzable algorithms, such as unit clause propagation[6] find SAT solutions with a finite probability[57]. Upper bounds use the fact that the probability of finding a satisfying assignment is bounded by the expected number of solutions. Refined versions of this argument[41, 90] partially eliminate the potentially high degeneracy of some solutions[82]. Both methods have been applied to $(2+p)$-SAT in [1], yielding the lines plotted in Fig 7.5.

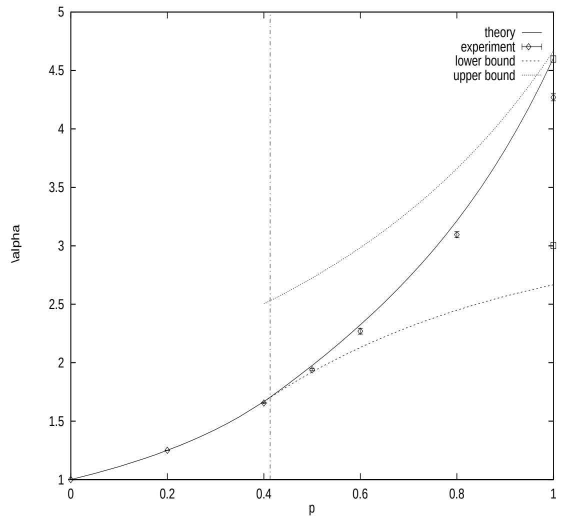


Figure 7.5: Theoretical and experimental results for the SAT/UNSAT transition in the 2+p-SAT model. The vertical line at $p_0$ separates the continuous transition from the discontinuous one. The full line is the replica-symmetric theory's predicted transition, believed exact for $p < p_0$, and the diamond data points with error bars are results of computer experiment and finite-size scaling. The other two lines show upper and lower bounds obtained in [1], while the stronger upper bound due to [90], and the best known lower bound, due to [57], are indicated by square data points.

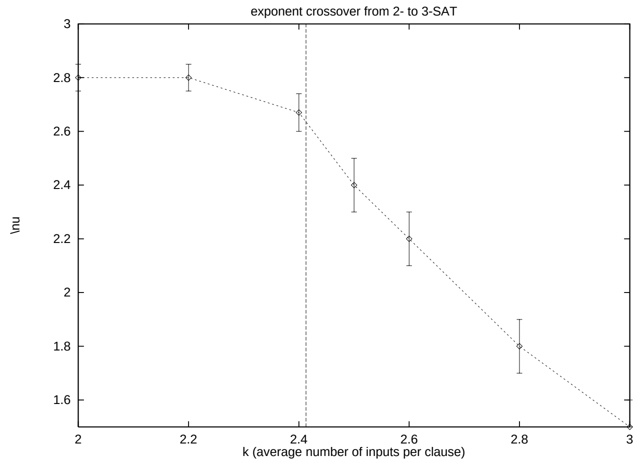


Figure 7.6: Crossover seen in the exponent $\nu$ governing the width of the critical regime, as $k$ increases from 2 to 3.

In figure 7.5, we show values of $\nu$ obtained from finite-size scaling analysis, which was obtained by plotting all quantities against the rescaled variable

$$y = N^{\frac{1}{\nu}}(\alpha - \alpha_c(k, N))/\alpha_c(k, N), \tag{7.21}$$

and making no assumptions about the $N$-dependence of $\alpha_c(k, N)$. Below $p_0$, the exponent $\nu$ is roughly constant and equal to 2.8, the value found for 2-SAT. This indicates that the critical behavior along the second order transition line in Fig. 1 is dominated by the 2-clauses in the formulae. When we include corrections to scaling in $y(\alpha, N)$ and in the probability of UNSAT, we find that $\nu$ may be as large as 3, the value that occurs in the percolation transition for undirected random graphs[17]. Since the UNSAT phase for $k = 2$ is one in which "constraint loops" become so ubiquitous that almost certainly there is some literal that implies its converse, we propose that the 2-SAT transition results from percolation of these loops, and is in the same universality class as random graph percolation, differing only in the corrections to its scaling behavior.

Above $p_0$, $\nu$ drops rapidly to 1.5. For $K \gg 3$, the values of $\nu$ tend to 1.0, a result that can be understood in the "annealed" limit discussed in [110],[82].

It is surprising that finite size scaling holds in the presence of discontinuous behavior of the order parameter characterizing the backbone. But this discontinuity is accompanied by smooth behavior of other thermodynamic quantities, e.g., entropy. First order transitions in pure solids involve two (or a finite number of) phases and do not exhibit critical fluctuations or scaling laws with non-integer exponents. The transition taking place for $p > p_0$, into an infinite number of distinct ground states, displays features of both first and second orders. This mixed behavior has recently been observed in random-field models [168], but has not been explained in detail yet.

Previous work showed that the cost of running the best heuristics, depth-first search algorithms with backtracking [37] increased exponentially with $N$ for any value of $\alpha$ for $k = 3$, with a pre-factor that could be mapped into a universal function by plotting it as a function of $y$[159]. The cost was maximized at $\alpha_{.5}(k, N)$, so we have obtained cost data at this value of $\alpha$ for $p = 0., .2, .4$ and $.6$ over the range of N that could be searched. The plot in Fig. 7.7 shows that the median cost increases linearly with N for $p < p_0$, while it increases dramatically over a smaller range of $N$ for $p > p_0$. Fig. 7.8 confirms that this increase is exponential already for $p = 0.6$.

Discontinuous nucleation of UNSAT regions due to the breakdown of replica symmetry and the "backbone" of frozen spins conveniently explain the apparently inevitable high cost of heuristic search near the phase boundary. Heuristics that proceed by "asserting" the possible value of a spin make early mistakes by mis-asserting a backbone spin, and take a long time to backtrack to correct their mistakes. Even if the backbone can be identified before the depth-first search begins, the problem that remains is one of organizing the search over the remaining spins that lie on the boundaries of "nuclei" or partial solutions to find the lowest energy arrangements of the whole solution, also involving much wasted effort.

The experiments confirm that the appearance of the backbone is discontinuous in Fig. 7.9. Above the threshold, the fraction of frozen spins found in small samples by exhaustive enumeration is relatively insensitive to $N$, especially for $k = 3$. At and below the threshold, the fraction of frozen spins decreases rapidly with increasing $N$. While the samples that could be studied are too small to permit extrapolation, the results are consistent with $< f >$ vanishing below $\alpha_c$. When the same calculation was carried out for 2-SAT, $f_c(2, \alpha)$ vanishes with increasing $N$ as $N^{-\delta}$ at threshold, with $\delta \simeq 0.37$, an exponent compatible with the value $1/\nu$ obtained from simple scaling arguments. In addition, the backbone fractions are differently distributed in the two cases. For $k = 3$, we find that a significant fraction of samples have nearly all of their spins "frozen," with the remainder contributing a tail with a small fraction of frozen spins. This two-peaked distribution is not observed for $k = 2$. Rather, the backbone seen at finite $N$ close to the threshold is due to finite size effects.

### 7.5.3 Discussion

The existence of a "backbone" has also been reported in the traveling salesman problem[89], with only a few bonds differing in many near-optimal tours. This observation has recently been turned to advantage by heuristics[157] that identify the backbone links and concentrate attention on the small subproblems that remain. Seperating the reducible from the irreducible part in frustrated systems was also suggested in [133], and may prove to be a generally valid

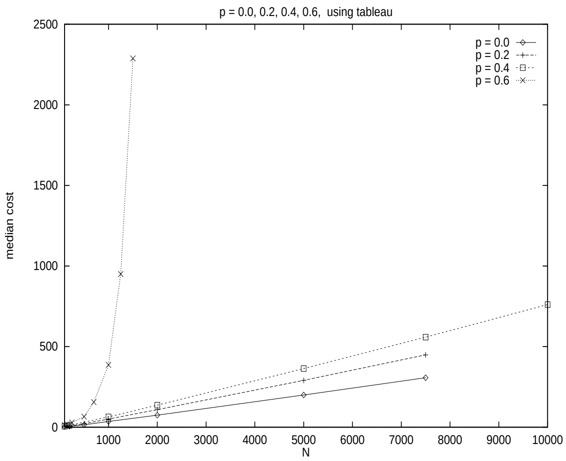


Figure 7.7: Median computational cost of proving a formula SAT or UNSAT using the TABLEAU search method, for $p$ ranging from 0 to 1. The data is is plotted on a linear scale, appropriate for the cases with $p < p_0$. Crossover seen in the exponent $\nu$ governing the width of the critical regime, as $k$ increases from 2 to 3.

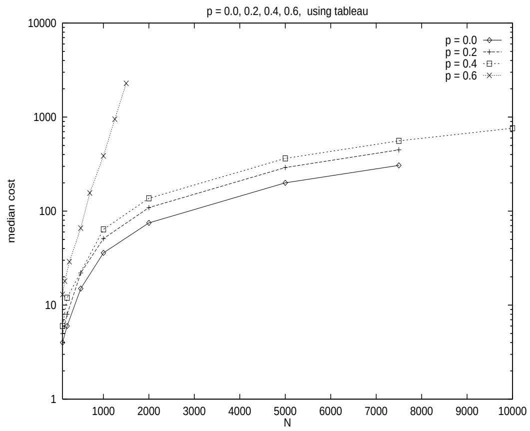


Figure 7.8: Median computational cost of proving a formula SAT or UNSAT using the TABLEAU search method, for $p$ ranging from 0 to 1. The semi-log scale scale show an exponential dependence of cost on $N$ for $p > p_0$.

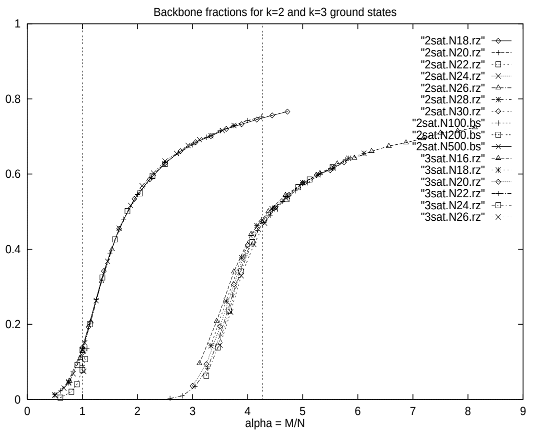


Figure 7.9: Backbone fractions as a function of $\alpha$ extracted from many samples for $k = 2$ and $k = 3$ cases with $N = 18$ to 30. The vertical lines mark the SAT/UNSAT thresholds in the limit $N \to \infty$. For 2-SAT, data obtained from larger sizes $N = 100, 200, 500$ show that the backbone fraction at the threshold decreases to zero.

approach. Efficient means of finding the backbone, or irreducible part, will be specific to each problem type, but should nonetheless provide a step ahead in algorithm efficiency.

Where else can the insights from k-SAT be applied? Another problem reporting a "backbone" is rigidity percolation on random lattices[78, 115]. Those models also exhibit simultaneous first and second order effects, discontinuous onset of order with non-trivial power law scaling. Many spin glass models are known to have first order transitions in their order parameters, but these have not been accessible to large computer experiment to test, e.g., for finite-size scaling. Granular materials, such as sand piles, are the 3D systems that rigidity percolation is intended to model. We suspect that they may have the same tendency to exhibit discontinuous and fractal behavior in a single transition.

## 7.6 Summary

In this chapter we have analyzed several properties of a large, random typical k-CNF formula. We developed a method for obtaining the shape of the threshold and a formula for the shape of the threshold in the large-k limit. The results was compared with a re-scaled version of the shape of the threshold found in numerical simulations, and they are found to be in very good agreement. We described a statistical mechanics formulation for the satisfiability problem, which is used in order to characterize the properties of typical large k-CNF formulae. An order parameter that measures the average overlap between satisfying assignments was derived, using the second moment of the number of satisfying assignments, and its rate of change with respect to $\alpha$ was used in order the predict the value of the threshold. The results are in good agreement with experimental results for $k \geq 3$.

We introduced a model that smoothly interpolates between the always-tractable 2-SAT problem and the intractable (in the worst case) 3-SAT problem: the 2+p-SAT model, in which an ensemble of CNF formulae with fraction p of 3–clauses and fraction $1 - p$ of 2–clauses is constructed. We found that, up to a critical $p_0$, statistical mechanics characteristics of the problem are similar to those of 2-SAT, while above this critical $p_0$, the problem has the same characteristics as the 3-SAT problem. Simulations results show, indeed, that at the critical $\alpha$, typical 2+p-CNF formulae are tractable up to $p_0$ and are intractable above $p_0$. The analysis suggests a possible explanation for the origin of intractability: for $p > p_0$ and $\alpha > \alpha_c$ all the solutions ("ground states") of a typical formula contain a "backbone": a set of variables whose value have to be same in the solutions. The existence of such a backbone poses a difficult problem for various heuristics for solving k-SAT problems.

# Part IV

# Learning of Sound Localization Using Direction-Dependent Spectral Data

# Chapter 8

# Learning of Sound Localization Using Direction-Dependent Spectral Data.

*"Sometimes a scream is better then a thesis."*

*R. W. Emmerson*

One of the most interesting aspects of learning theory is its ability to shed some light on learning processes in the most complex system known: the brain. Since real neuronal circuits are, in general, tremendously complex, theoretical models of learning in the brain are often merely a caricature of real-life processes. Still, such models help us to gain further understanding regarding the structure and functions of neuronal circuits, and they also contribute to the de-mystification of mental processes.

In this part of the study, we employ learning methods of artificial neural-networks in order to show that a rather complicated problem, the estimation of the direction of an unfamiliar sound source based on monoaural cues, can be handled by simple neural circuits.[1]

## 8.1 Introduction

The major cues for sound localization are Interaural Time Differences (ITD) and Interaural Intensity Differences (IID) between the two ears. However, these cues have approximately the same values for locations on "cones of confusion" [15] and in particular cannot be used for determining the elevation of sound sources located on the medial saggital plane.

The pinnae, head and torso modify the spectrum of the sound signal; this distortion, which is direction-dependent, is quantified by measurement of the transfer function from free-field to ear drum, known as the *Head Related Transfer Function* (HRTF). It is accepted today that the spectral features introduced by the HRTFs affect sound localization [107] [117] and

[1]The work described in this chapter was done in collaboration with Dr. Israel Nelken from Hadassah Medical School and Daniel Gill from the computer science department in the Hebrew University. The work is now developed to a journal paper.

may serve as cues for disambiguation of locations on the cones of confusion.

The use of HRTFs for sound localization poses a challenge to the auditory system: spectral features of a sound at the ear drum can be either properties of the sound source itself, or properties of the HRTF. Thus, in order to extract localization information from the monaural spectrum, the auditory system must be able to distinguish between the source spectrum and the HRTF. Previous attempts to model the use of spectral information for sound localization did not address this fundamental problem: Neti *et al.*[124] built a neural network model that uses spectral Interaural intensity difference (IID) cues in both monoaural and binaural form. However, their approach to the use of HRTF amplitude spectra assumed a flat spectrum source, which is not a realistic scenario. Nandy and Ben-Arie [119] presented a neural network that extracts IID spectral patterns derived from binaural HRTFs as an internal IID spectrum : a feed-forward network maps a HRTF dependent interaural IID spectrum onto a two dimensional grid that represents the auditory space map with elevation and azimuth dimensions. Their network can use the additional spectral-dependent information in order to improve the azimuthal and elevational allocation of noisy versions of *familiar sounds.*

In this work we present a neural network model motivated by biological considerations for the hardest localization problem, namely elevational allocation of unfamiliar natural sound signals, based solely on monoaural direction-dependent spectral information. The resulting network was able to determine the elevation of an unfamiliar sound to within $\pm 3.5°$ in all of the test cases.

## 8.2 Description of the model

The main goal of this work was to construct a model that, on the one hand will be as simple as possible, yet on the other hand, will capture the essential features of the biological mechanisms that might be involved in the determination of the elevation of an *unfamiliar* natural sound signal, based on monoaural cues. In order to find a simple circuit that can solve this problem with reasonable accuracy, a comparative study of several models, based on simulated neural circuits was performed.

### 8.2.1 Source signal spectra:

The source signals consisted of a set of natural sounds (birds chirping, dogs barking, cows mooing, etc.). The signals were sampled at 48 kHz and transformed into the frequency domain. The spectrum was then represented using 300 bins of 100 Hz width each. Some typical signals are shown in figure 8.1.

### 8.2.2 Pre-processing of source spectra

The spectral properties of the stimulus signal are modulated by the Head-Related-Transfer-Function, which depends on the relative direction of the source. HRTFs for 17 relative

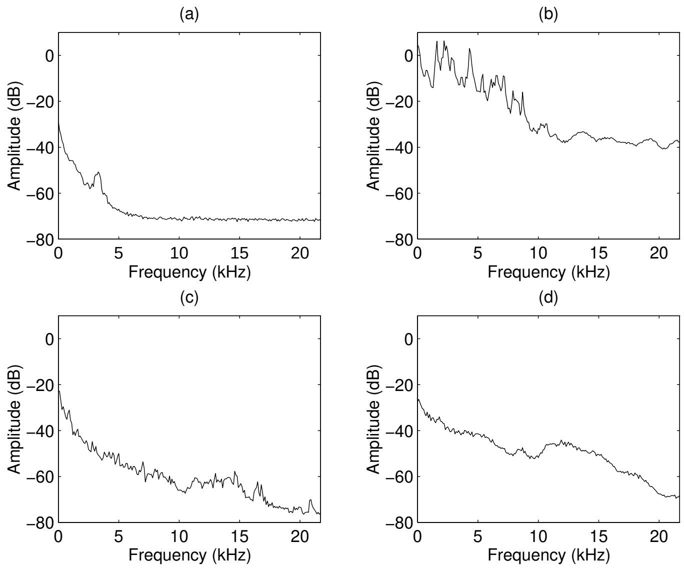


Figure 8.1: Typical power spectra of the free-field signals : (a) & (b) are taken from the training set, (c) & (d) are from the test set. It can be seen that the signals are rather different from each other.

elevations, from $-30°$ to $90°$ with a resolution of $7.5°$ that were measured on a cat were used. (The HRTF data is courtesy of E. Young[185]) HRTF curves for two relative elevations, $+7.5°$ and $-7.5°$ are shown in figure 2.

The HRTFs at a relative elevation $\varphi$ were used as linear filters $H(\omega, \varphi)$. Given a free-field signal $X(\omega; t)$, the stimulus spectrum at the ear drum can be expressed as:

$$Y(\omega, \varphi; t) = H(\omega, \varphi)X(\omega; t) \tag{8.1}$$

The resulting spectrum was then compressed logarithmically, to mimic the approximate logarithmic dependence of auditory nerve firing rates on sound levels.

The next step in the processing was motivated by known physiological properties of Dorsal Cochlear Nucleus (DCN) type IV neurons and their responses to wide-band sounds. These neurons, which are spontaneously active, are weakly activated by wide-band noise bands. However, type IV neurons are strongly inhibited by wide-band noise bands when a spectral notch is introduced near their best frequency (BF). This property, and the fact that spectral notches are a major feature of cat HRTFs [143], led to the suggestion that type IV neurons participate in the extraction of spectral cues for sound localization [124]. To mimic these type IV neurons properties, a linear notch detector was created by a center-surround type organization of weights, as shown in Fig. 3. The total response of the neuron to "white-noise" was a small, excitatory response. When a notch occurs in the spectrum near the BF of the neuron, the strong central excitation is absent and the neuron is not activated at all. Denote by $f_i(\omega)$ the frequency response function of the i-th type IV neuron, the response of the neuron due to the a signal $X(\omega_i; t)$ from a relative direction $\varphi$ can be written as:

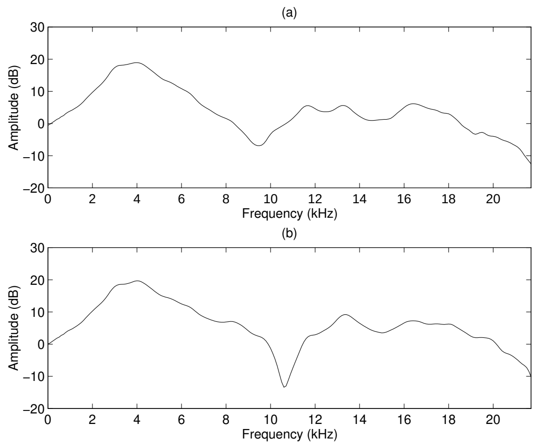


Figure 8.2: HRTF for $-15°$ (a) and $+15°$ (b).

$$R_i(\varphi;t) = \int_0^{f_{max}} f_i(\omega) \log |H(\omega,\varphi)X(\omega;t)|^2 d\omega. \tag{8.2}$$

An array of $N$ such notch detector neurons with a typical notch-width of about 500 Hz and best frequencies spaced logarithmically in the range of 5-21 kHz was used in order to sample the filtered spectra. This scheme greatly reduced the dimensionality of the problem - nine such neurons were enough in order to extract the relevant direction-dependent information and to assure a good performance. That such a sparse spectral representation is sufficient for the localization of unfamiliar sounds is an indication of the large amount of localization information available in the spectral cues. Further improvement in network performance was achieved when the average input response (over the various “snapshots”) was subtracted from the input-layer responses.

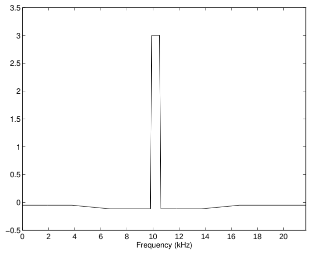


Figure 8.3: A notch detector

### 8.2.3 Network architecture and training.

Several candidate architectures were considered, in order to find a network with a minimal complexity that could adequately perform the localization task. A three layer feed-forward network gave the optimal results: the input layer consisted of $N$ notch-detector neurons, as discussed above. The output of the input layer was fed into a "hidden layer" of 9 neurons, using a set of adjustable connections ("synapses") and their outputs were fed into an output layer of 15 neurons (one for each of the reference elevations, excluding the extreme elevations). The network was implemented using the MATLAB Neural Networks Toolbox. A scheme of the network architecture is shown in figure 4.

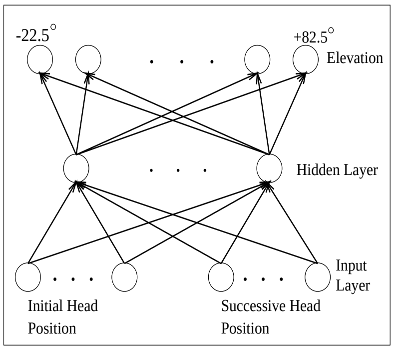


Figure 8.4: Basic network architecture with two head positions.

The network was trained to give an output of 1 in the neuron that corresponded to the label of the true elevation, output of 0.3 for the neurons that corresponded to the nearest elevations (true elevation $\pm 7.5°$) and an output 0 in all the other neurons. The method of training was back-propagation, using "momentum" and an adaptive learning-rate.

### Finding the elevation of a familiar signal

When learning to localize *familiar* sounds, each signal out of a training set of 45 signals, was presented to the network in each of the 17 elevations, and the network was trained until the training error was practically zero, meaning that all the familiar signals were localized correctly. This task seems to be relatively easy. Familiar sounds can indeed be localized better than unfamiliar sounds on the median plane [15].

### Finding the elevation of an unfamiliar signal

When learning to localize the elevation of an *unfamiliar signal*, the signal should be presented to the network more than once, in order to distinguish the elevation-dependent information from the spectral variations in the source spectrum. This can be achieved by binaural presentation of the stimulus; since, in general, the HRTF in the two ears are different, comparing the spectra in the two ears can be used to identify features of the source spectrum. However, because the elevation discrimination is on the median plane, the HRTFs of each location on the median plane to the two ears are almost identical (due to head symmetry). Therefore, we assumed that the cat uses two or more consecutive "snapshots" of the stimulus after making a small head movement. For this purpose, we use 1-2 additional transformed replicas of the original signal input layer as the signal that would be heard in the new position of the head. One replica corresponds to the original elevation, one to an elevation of $+7.5°$ and one (optional) to an elevation $-7.5°$. cf. fig. 4. Note that since a cat can move its pinnae, multiple snapshots could be achieved by positioning the pinnae in slightly different elevations and using the sounds reaching the two ears as two separate snapshots.

Subtracting of the average input response from the input-layer response improved the network performance and increased the *learning rates*. The network was trained using 3 different signals in each of the 15 elevations from $-22.5°$ to $+82.5°$.

The most interesting aspect of the network performance is its ability to estimate the relative elevation for a novel input signal. After the training stage, the network performance was checked over a set of 45 different natural sound signals, with rather different spectral properties. The output of the network was considered to be the index of the neuron with the strongest response (A "winner-takes-all" scheme). The output of the network for novel stimulus was, in general, close to "1" in the neuron that corresponds to the true elevation, and close to "0" in all the other neurons.

The "generalization error" of the network was considered to be the relative number of times in which the network predicted a wrong label for the elevation. When information from three consecutive elevations was used, performance was error-free. When information from only two consecutive directions was used, the error increased to 40%, but most of these errors occurred between two consecutive directions, which means that the basic effect is a deterioration of the resolution. Interestingly, the error variance for humans detecting elevation of a sound source on the median plane is about $10^{circ}$, comparable with the errors committed by our network using two snapshots.

The network was able to learn the task using a surprisingly low number of training samples (only 3 training samples per direction). This is probably due, in part, to the efficient biologically-motivated preprocessing scheme, which made it possible to achieve a significant dimensionality reduction by sampling the spectrum with only 9 notch detectors. In addition, we believe that an efficient internal representation was created in the hidden layer, since detection of each of the 15 possible elevations was made with respect to the same network; thus, we may have benefited from "multi-task learning" [22]. A similar setting, with one output unit,in which each elevation was trained separately,yielded much poorer results.

## 8.3 Discussion

Although the use of notch detectors was motivated by biological results, the network is far from being biologically realistic. First, all our input neurons had the same bandwidth, independent of their BF; this is in marked contrast to the biological finding that bandwidth increases roughly in proportion to BF. It was however shown by Nelken and Young [123] that the narrowest notch that inhibits a type IV neuron is only weakly correlated with BF; thus, for the exclusive goal of creating notch detectors, the use of constant bandwidth filters may not be an unacceptable approximation. Second, the coverage of the spectrum was very sparse. Thus, our results should be taken as a lower bound on the amount of localization information that can be extracted from spectral cues independently of source spectrum. The high amount of information available even with such a sparse sampling of the spectrum indicates that 2 snapshots are more than enough to achieve a reasonable localization capability based on spectral cues. Within the context of these limitations, some rather interesting insights regarding the process of monoaural localization can be obtained:

- Spectral cues, based on HRTF can provide sufficient information for determining the elevation of unfamiliar sound-sources. Furthermore, this information can be extracted by simple neural circuits, with a small number of components.
- The sample complexity (i.e. the number of labeled samples that are required) and the computational complexity of spectral localization can be made surprisingly low, provided that proper pre-processing is performed.
- Multi-task learning (i.e. learning to recognize the various elevations using the same input layer and hidden layer) might have contributed to the success of this network, by constructing an efficient internal representation in the hidden layer.

# Part V

# Appendix

# Chapter 9

# Appendix

## 9.1 Some mathematical essentials

### 9.1.1 Pseudo-metric spaces, packing and covering numbers

Given a set of points $T$, we say that there exists a *metric structure* over $T$ if there exists a function $d$, which is defined for all the possible pairs of points $t_i, t_j \in T$, such that:

1. $d(t_i, t_j) = d(t_j, t_i), \ \forall t_i, t_j \in T$
2. $d(t_i, t_j) \leq d(t_i, t_k) + d(t_k, t_j), \forall t_i, t_j, t_k \in T$
3. $d(t_i, t_j) \geq 0 \ \forall \ t_i, t_j \in T$ and $d(t_i, t_j) = 0$ if, and only if $t_i = t_j$.

$d(t_i, t_j)$ is called the *distance* between $t_i$ and $t_j$.

If the conditions above hold except that there exist points in $T$ such that $d(t, t') = 0$ and $t \neq t'$, then we say that $d$ is a *pseudo-metric* over $T$.

The set of points, together with the metric (pseudo-metric) function is called a *Metric (Pseudo-Metric) Space.*

Given a metric (pseudo-metric) space, it is sometimes useful to consider a subset of the points of this space which is representative in the sense that each point $t$ in the space has at least one point $t'$ in the representative set such that $d(t, t') < \epsilon$ for some $\epsilon > 0$. The representative set is called the $\epsilon-$cover of the space. formally:

**Definition 4 ($\epsilon$-cover)** *Let $d$ be a (pseudo-)metric over a space $\Omega$ and let $T \subseteq \Omega$. For $\epsilon > 0$ a set $N \subset \Omega$ (not necessarily contained in $T$) is an* $\epsilon$-cover [1] *of $T$ if for all $\bar{x} \in T$ there is a $\bar{y} \in N$ with $d(\bar{x}, \bar{y}) \leq \epsilon$. The function $\mathcal{N}(\epsilon, T)$ denotes the size of the smallest $\epsilon$-cover for $T$. We refer to $\mathcal{N}(\epsilon, T)$ as a* covering number. [2]

[1] In Vapnik's book the terms $\epsilon-$net is used instead of $\epsilon-$cover. However, the term $\epsilon-$net has a somewhat different meaning in the more recent learning theory terminology.

[2] Note that $\mathcal{N}(\epsilon, T)$ is finite whenever $T$ is compact.

$\epsilon-$cover have a major role in the the proofs of PAC-learnability[16].

### 9.1.2 Bounds on deviations from the mean

#### Markov inequality

Let $x$ be a positive random variable with expectation $\mu = E(x)$. Then for any $\lambda \geq 0$ the following inequality holds:

$$Pr(x \geq \lambda) \leq \frac{E(x)}{\lambda}. \tag{9.1}$$

#### Chebyshev inequality

let $x$ be a random variable with expectation: $\mu = E(x)$

$$Pr(|x - \mu| > \epsilon) \leq \frac{\sigma^2}{\epsilon^2}, \quad \forall \epsilon > 0. \tag{9.2}$$

Applying Chebyshev inequality to: $y = \frac{1}{n}\sum_{i=1}^{n} x_i$. we obtain:

$$var(y) = E\left[(y - E(y))^2\right] = \frac{1}{n^2}\sum_{i=1}^{n} var(x_i) = \frac{\sigma^2}{n} \tag{9.3}$$

$$\Longrightarrow Pr\left[\left|\frac{1}{n}\sum_{i=1}^{n} x_i - E(x)\right| \geq \epsilon\right] \leq \frac{\sigma^2}{n\epsilon^2} \longrightarrow 0 \quad \forall \epsilon. \tag{9.4}$$

.

#### Chernoff and Hoeffding bounds

Let $X_1, \ldots X_m$ denote the outcomes of $m$ independent Bernoulli trials, with
$\mathbf{Pr}\left[X_i = 1\right] = p, ;\ \mathbf{Pr}\left[X_i = 0\right] = 1 - p$. Let $S = X_1 + X_2 + \ldots X_m$.
Then: $E\left[S\right] = E\left[X_1\right] + \cdots + E\left[X_m\right] = pm$ The Chernoff and Hoeffding bounds state that the probability that $S$ deviates from its mean $pm$ by an amount that decreases exponentially with $m$:

- *Additive Form (Hoeffding bound)*

$$\mathbf{Pr}\left[|S - pm| > \gamma m\right] \leq 2e^{-2m\gamma^2}$$

- *Multiplicative Form (Chernoff bound)*

$$\mathbf{Pr}\left[S/pm > 1 - \gamma\right] \leq e^{-mp\gamma^2/3}$$

$$\mathbf{Pr}\left[S/pm < 1 - \gamma\right] \leq e^{-mp\gamma^2/2}$$

## 9.2 Permanent uncertainty: on the quantum evaluation of the determinant and the permanent of a matrix


### abstract

We investigate the possibility of evaluating permanents and determinants of matrices by quantum computation. All current algorithms for the evaluation of the permanent of a real matrix have exponential time complexity and are known to be in the class $P^{\#P}$. Any method to evaluate or approximate the permanent is thus of fundamental interest to complexity theory. Permanents and determinants of matrices play a basic role in quantum statistical mechanics of identical bosons and fermions, as the only possible many particle states made of single particle wave functions.

We study a novel many-particle quantum-computational model in which an observable operator can be constructed, in polynomial-time complexity, to yield the determinant or the permanent of an arbitrary matrix as its expectation value. Both quantities are estimated, in this model, by quantum mechanics systems in a polynomial time, using fermions and bosons respectively. It turns out that the *variance* of the measurements in a "noise-free case" is zero for the determinant and exponential (in the rank of the matrix) for the permanent. The intrinsic measurement variance is therefore the quantum mechanical correspondence to the computational complexity gap.


### 9.2.1 Introduction

Quantum computational models have recently become important for the theory of computational complexity when it was shown that factorization can be performed in a polynomial time by a Quantum Turing machine[163].

An intriguing computational question is naturally raised when considering quantum statistical mechanics of identical particles. By Pauli's "spin-statistics" theorem[131], the quantum state of identical particles is in general either symmetric with respect to exchange of particles for integer spin particles (bosons), or anti-symmetric for half-integer spin particles (fermions). The only possible completely symmetric or anti-symmetric combinations of general single particle functions are the permanent or determinant of those functions, respectively.

Whereas in QM the difference between permanents and determinants simply corresponds to the choice between the quantum statistics of bosons and fermions, determinants and permanents have very different computational complexity by all known algorithm. While the determinant is invariant to matrix similarity transformations and can thus be easily evaluated in polynomial time in the size of the matrix, no such polynomial time algorithm is known for the permanent. Moreover, there is no known algorithm for the evaluation of the permanent that can avoid the summation over exponential many terms, i.e. the permanent is known to be in the complexity class $\#P$-complete. In this paper we would like to understand the quantum mechanical correspondence to this dramatic computational complexity gap. We approach this problem by considering many particle quantum observable that correspond to the determinant of a given matrix in the fermion case, and to the permanent of the matrix in the boson case.

The permanent of a matrix $\mathbf{A} = [a_{ij}]$ is defined as:

$$\mathrm{per}[\mathbf{A}] \equiv \sum_{\{\mathrm{i_1..i_n}\}} \mathrm{a_{1i_1} \cdot a_{2i_2} ... a_{ni_n}} \ , \tag{9.5}$$

where the set of indices $\{(i_1..i_n)\}$ denotes the set of all the permutations of $1...n$ [24]. The determinant is defined by the same sum of the permutations, where in addition each odd permutation is taken with a negative sign.

Permanents occur naturally in various counting problems in combinatorics[108], graph theory[21], and logic. If there was a way to evaluate, or even to approximate permanents in a polynomial time, then *every* $P^{\#P}$ (and thus every NP-complete) problem could have been evaluated in polynomial time [174] [61]. It can be shown that even the approximation of the log of the permanent is a hard problem[188]. The key technical reason for that computational difference between permanents and determinants is that determinants are invariants under matrix similarity transformations of the matrix, while no such simple invariants are known for the permanent. So diagonalization of the matrix or similar transformations can not help for permanents.

Determinants and permanents are fundamental functions in quantum statistics of identical particles, as described below.

- The wave-function of indistinguishable fermions can be written in the Slater-determinant form. Namely, if there are $n$ fermions in $n$ single-particles states, $|1\rangle...|n\rangle$, than the $n$-Particles wave-function is given by,

$$\begin{aligned} |\Psi_A(1...n)\rangle &= \frac{1}{\sqrt{n!}}[|1,1\rangle|2,2\rangle...|n,n\rangle \\ &- |1,2\rangle|2,1\rangle...|n,n\rangle + ...] \\ &= \frac{1}{\sqrt{n!}} \sum_{\{i_1,...i_n\}} (-1)^{\pi}|1,i_1\rangle \\ & |2,i_2\rangle...|n,i_n\rangle = \frac{1}{\sqrt{n!}}\det[|i,j\rangle]. \end{aligned} \tag{9.6}$$

  The sum is taken over all the permutations, $\pi$, of $1 \ldots n$ , where the first index denote the state and the second denote particles ($i_j$ is the permutation index).

- Particles with an integer spin ("bosons") obey the Bose-Einstein statistics: the many-particles wave-function is completely symmetric under exchange of any two particles,

$$
\begin{aligned}
|\Psi_S(1...n)\rangle &= \frac{1}{\sqrt{n!}}[|1,1\rangle|2,2\rangle...|n,n\rangle \\
&+ |1,2\rangle|2,1\rangle...|n,n\rangle + ...] \\
&= \frac{1}{\sqrt{n!}} \cdot \sum_{\{i_1,...i_n\}} |1,i_1\rangle|2,i_2\rangle...|n,i_n\rangle \\
&= \frac{1}{\sqrt{n!}} \cdot \mathrm{per}\left[|\mathrm{i},\mathrm{j}\rangle\right]. \qquad (9.7)
\end{aligned}
$$

In what follows we address two natural questions: (a) is there a quantum mechanical observable (or measurement) that can be constructed in order to evaluate the permanent or determinant of an arbitrary matrix; (b) what is the difference between the evaluation of determinants and permanents from the quantum computation point of view.

### 9.2.2 Measuring the permanent of an arbitrary Hermitian matrix

Let $\mathbf{A} = [a_{ij}]$ denote the Hermitian matrix whose permanent is required. If $\mathbf{A}$ is not Hermitian it can always be installed into an Hermitian matrix $\tilde{\mathbf{A}}$ of the form,

$$
\tilde{\mathbf{A}} = \begin{pmatrix} \mathbf{0} & \mathbf{A} \\ \mathbf{A}^T & \mathbf{0} \end{pmatrix} \qquad (9.8)
$$

and for this matrix, $\mathrm{per}[\tilde{\mathbf{A}}] = \mathrm{per}[\mathbf{A}])^2$. So every permanent of a real matrix can be evaluated, up to sign, by the permanent of an Hermitian matrix.

Let $\omega$ be any single particle QM observable, with continuous unbounded spectrum, such as the linear momentum operator $\mathbf{p}$.

We denote by

$$
\Omega = \prod_{i=1}^{n} \omega_i \ , \qquad (9.9)
$$

the product of these single particle observables, which is now a many particle observable on its own. Since single-particle operators that operate on different particles are always commutative the order of the operators in the product is not important.

The Hermitian matrix $\mathbf{A}$ can be diaganolized by a unitary transformation,

$$
\mathbf{U}^\dagger \mathbf{A} \mathbf{U} = \mathbf{D} = \begin{pmatrix} \lambda_1 & 0 & 0 & \dots \\ 0 & \lambda_2 & 0 & \dots \\ 0 & 0 & \lambda_3 & \dots \end{pmatrix} \ , \qquad (9.10)
$$

where $\mathbf{U} = [u_{ij}]$ is a unitary matrix, $\mathbf{U}^\dagger \mathbf{U} = \mathbf{I}$.

We denote by $|\lambda_j\rangle$ the eigenstate of the single particle observable $\omega$ that corresponds to the eigenvalue $\lambda_j$,
$\langle\lambda_i\lambda_j\rangle + \delta_{ij}$.

The idea is to prepare the many particle system in a special superposition, that depend on the given matrix $\mathbf{A}$, and then perform a measurement of the product operator $\Omega$. We prepare the many particle system such that there is a single boson/fermion in each of the following single-particle superposition of the eigenstates of $\omega_{\mathbf{i}}$,

$$|i\rangle = \sum_l u_{li}|\lambda_l\rangle \ . \tag{9.11}$$

In that state the matrix elements of $\omega$, in this single particle basis, are given by,

$$\langle i|\omega|j\rangle = (\sum_l \langle\lambda_l|u_{il}^*)\omega(\sum_k u_{kj}|\lambda_k\rangle) =$$
$$(\sum_l \langle\lambda_l|u_{il}^*)(\sum_k \lambda_k u_{kl}|\lambda_k\rangle) = \sum_k u_{ik}^* u_{kj}\lambda_k = a_{ij} \quad . \tag{9.12}$$

The new basis is orthonormal due to the unitarity of $\mathbf{U}$. We denote by $|\Psi\rangle$ the $n$-particle state of the above single particle superpositions. Due to the indistinguishability of the particles, $|\Psi\rangle$ must be symmetric/antisymmetric with respect to particle permutations.

The expectation value of the operator $\Omega$ in the n-particle state $|\Psi\rangle$ is given, for the boson case, by:

$$\begin{aligned} &\langle\Psi|\Omega|\Psi\rangle = \\ =\ & \frac{1}{n!}\cdot\left(\sum_{\{i_1..i_n\}}\langle 1,i_1|\ldots\langle n,i_n|\right)\left(\prod_{i=1}^{n}\omega_{\mathbf{i}}\right)\left(\sum_{\{i_1...i_n\}}|1,i_1\rangle\ldots|n,i_n\rangle\right) \\ =\ & \frac{1}{n!}\cdot n!\cdot\sum_{\{i_1...i_n\}}\langle 1|\omega|i_1\rangle...\langle n|\omega|i_n\rangle = \mathrm{per}[\mathrm{A}] \ . \end{aligned} \tag{9.13}$$

Similarly, this expectation is det[A] for the fermion case.

For illustration, consider a $2\times 2$ matrix. The symmetric two-particles wave-function is,

$$|\Psi(1,2)\rangle = \frac{1}{\sqrt{2!}}(|1,1\rangle|2,2\rangle + |1,2\rangle|2,1\rangle) \ , \tag{9.14}$$

and the expectation value of $\Omega$ is:

$$\begin{aligned} \langle\Psi|\Omega|\Psi\rangle &= \frac{1}{2!}[(\langle 1,1|\langle 2,2| + \langle 1,2|\langle 2,1|)\omega_{\mathbf{1}}\omega_{\mathbf{2}}(|1,1\rangle|2,2\rangle + |1,2\rangle|2,1\rangle)] \\ &= \frac{1}{2!}[\langle 1|\omega|1\rangle\langle 2|\omega|2\rangle + \langle 2|\omega|1\rangle\langle 1|\omega|2\rangle + \langle 2|\omega|2\rangle\langle 1|\omega|1\rangle + \langle 1|\omega|2\rangle\langle 2|\omega|1\rangle] \\ &= [\langle 1|\omega|1\rangle\langle 2|\omega|2\rangle + \langle 2|\omega|1\rangle\langle 1|\omega|2\rangle + \langle 2|\omega|2\rangle\langle 1|\omega|1\rangle + \langle 1|\omega|2\rangle\langle 2|\omega|1\rangle] \\ &= a_{11}a_{22} + a_{12}a_{21} = \mathrm{per}[\mathbf{A}] \quad . \end{aligned} \tag{9.15}$$

Similar expression holds for the determinant, where all the antisymmetric (odd) permutations switch sign for fermions.

Thus, both the *permanent* and the determinant of an *arbitrary hermitian matrix* can be expressed as the *expectation value* of an n-particle quantum mechanical observable in a state of $n$ identical bosons or fermions.

The computation required for the preparation of the system are only the computations required for the diagonalization of the matrix $\mathbf{A}$ and the preparation of the single-particle superpositions, and therefore can be done in a polynomial number of steps. The evaluation of the determinant/permanent of the states is "done" by the spin-statistics by itself.

Quantum mechanical measurement can yield only one of the eigenvalues of the measured operator. Thus, measuring $\omega$ in a superposition of its eigenstates $|\lambda_1\rangle ... |\lambda_n\rangle$ can therefore yield one of the $n$ possible eigenvalues, $\lambda_1 .. \lambda_n$. Similarly, measuring the n-particle observable $\Omega = \prod_{i=1}^{n} \omega_{\mathbf{i}}$ in the state $|\Psi\rangle$ can therefore yield only one of the possible outcomes,

$$O_J \doteq \lambda_1^{n_1^J} \lambda_2^{n_2^J} ... \lambda_n^{n_n^J} \ , \tag{9.16}$$

where $n_i^J = 0, 1, ...n$ and for all $J$ and $\sum_{i=1}^{n} n_i^J = n$. The total number of possible outcomes of this measurement is

$$\binom{2n-1}{n} \approx 2^{2n} \ ,$$

i.e., the number of possible ways to distribute $n$ identical balls into $n$ distinct cells. The expectation value of $\Omega$ is obtained, asymptotically in the number of measurements,

$$\langle \Omega \rangle = \mathrm{per}[\mathbf{A}] = \sum_{J=1}^{\binom{2n-1}{n}} \lambda_1^{n_1^J} \lambda_2^{n_2^J} ... \lambda_n^{n_n^J} P(n_1^J ... n_n^J). \tag{9.17}$$

Whereas the permanent include $n! \approx \left(\frac{n}{e}\right)^n$ terms with equal weights, the decomposition in eq.9.17 contains only $\approx 2^{2n}$ different terms, each with a different weight. This decomposition can be considered as the probability-amplitude space induced by the permanent.

The probability to get the outcome

$$O_J \doteq n_1^J n_2^J ... n_n^J$$

in a measurement of $\Omega$ in the boson case, is given by,

$$\left| \frac{1}{\sqrt{n_1! .. n_n!}} \sum_{\{i_1 .. i_n\}} ((\langle \lambda_{1,,i_1} |)^{n_1^J} .. ((\langle \lambda_{n,i_n} |)^{n_n^J} |\Psi\rangle \right|^2 \ , \tag{9.18}$$

where $\langle \lambda_{1,i_1}|$ denotes that the $i_1$ particle is in the eigenstate $\langle \lambda_1|$, etc.

The composite $n$-particles wave-function $\Psi$ can be written as

$$\begin{aligned} |\Psi\rangle &= \frac{1}{\sqrt{n!}} \sum_{\{i_1 ... i_n\}} |1, i_1\rangle |2, i_2\rangle ... |n, i_n\rangle \\ &= \frac{1}{\sqrt{n!}} \sum_{\{i_1 .. i_n\}} \prod_{k=1}^{n} (\sum_{j=1}^{n} u_{kj} |\lambda_{j,i_k}\rangle) \ . \end{aligned} \tag{9.19}$$

Since:

$$\langle \lambda_{j,i_n} || \lambda_{l,i_m} \rangle = \delta_{jl} \delta_{i_n i_m} \ ,$$

we obtain sum of the permutations as,

$$\sum_{\{i_1 \ldots i_n\}} \frac{1}{\sqrt{n_1^J! \ldots n_n^J!}} (\langle \lambda_{1,,i_1} |)^{n_1^J} \ldots (\langle \lambda_{n,i_n} |)^{n_n^J}$$

$$(\sum_{j=1}^{n} u_{1j} | \lambda_{j,i_1} \rangle) \ldots (\sum_{j=1}^{n} u_{nj} | \lambda_{j,i_n} \rangle)$$

$$= \frac{1}{\sqrt{n_1^J! \ldots n_n^J!}} \mathrm{per}[\mathbf{U}^J] \ . \tag{9.20}$$

The matrix $\mathbf{U^J}$ is derived from the unitary matrix, $\mathbf{U}$, with the first row taken $n_1^J$ times, the second row $n_2^J$ times, etc. [108].

The probability amplitude for any of the outcomes of the measurement of the product is by itself proportional to a permanent of a matrix, and therefore a direct evaluation of the joint probabilities $P(n_1^J, n_2^J \ldots n_n^J)$ is a again in $P^{\#P}$, i.e., exponential in the rank of the matrix.

### 9.2.3 Physical implementation

While in general, the evaluation of an expectation value of an operator which is constructed from a product of many single-particle observables may be a difficult, an actual, rather simple physical device which allows the evaluation of permanents of $n \times n$ matrices can be constructed, using a scattering experiment with $n$ input channels and $n$ output channels, as schematically shown in figure 1. In this device, amplitude of the scattering from the i-th input channel to the $j$ output channel is just $u_{ij}$. If a single boson is enters each of the input

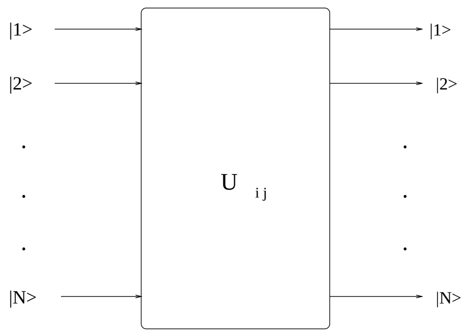


Figure 9.1: A sketch of the physical implementation.

channels, than each of the output channels can contain from 0 to $n$ bosons. The probability for each of these
$J = 1 \ldots \binom{2n-1}{n}$ possible scattering outcomes is given by,

$$P(n_1^J, n_2^J, \ldots n_n^J) = \frac{1}{\sqrt{n_1! n_2! \ldots n_n!}} \mathrm{per}[\mathbf{U^J}] \ . \tag{9.21}$$

If $M$ such scattering experiments are performed and for each outcome, $O_J$, the term $\frac{1}{M}\lambda_1^{n_1^J} \cdot \lambda_2^{n_2^J} ... \lambda_n^{n_n^J}$ is added, then in the limit $M \to \infty$ this sum converges, by the law of large numbers, to the expectation value, i.e., the real value of the permanent.

### 9.2.4 The accuracy of the measurement

As can be expected, while there is nothing so far that discriminates the permanent from the determinant in the many-particle quantum measurement, the caveat must be in the variance, or precision of our estimate. The accuracy in which the above procedure yields an estimate to the value of the permanent, in a finite number of measurements, depends on the variance in the measurement, or the intrinsic quantum mechanical uncertainty of our product observable $\Omega$.

The second moment of $\Omega$, in the many-particle state $|\Psi\rangle$ is given by,

$$\langle \Omega^2 \rangle = \langle \prod_{i=1}^{n} \omega_i^2 \rangle = \text{per}[\mathbf{B}]. \tag{9.22}$$

where $\mathbf{B}$ is the matrix whose elements are:

$$b_{ij} = \langle i|\omega^2|j\rangle.$$

However, this expectation value is simply related to the original matrix, $\mathbf{A}$, through,

$$\begin{aligned}[\mathbf{A}^2]_{ij} = \sum_l a_{il} a_{lj} = \sum_l \langle i|\omega|l\rangle\langle l|\omega|j\rangle = \\ = \langle i|\omega^2|j\rangle = b_{ij}\end{aligned} \tag{9.23}$$

and therefore,

$$\langle \Omega^2 \rangle_\Psi = \text{per}[\mathbf{A}^2] \ . \tag{9.24}$$

Since each of the elements of $\mathbf{A}^2$ is a sum of $n$ products of pairs of elements of the Hermitian matrix $\mathbf{A}$ – the permanent of $\mathbf{A}^2$ might be exponentially larger than $(\text{per}[\mathbf{A}])^2$. This means that the intrinsic variance can be exponential in the rank of the matrix, so an exponential number of experiments might be needed for a good approximation of the permanent.

If the same scheme is applied to fermions, the second moment in the measurements of $\Omega$ will be

$$\langle \Omega^2 \rangle_\Psi = \det[\mathbf{A}^2] \ . \tag{9.25}$$

However, since

$$\det[\mathbf{A}^2] = (\det[\mathbf{A}])^2$$

the variance in this measurements of $\Omega$ is identically *zero*. Thus for the fermionic case, the exact value of the determinant is obtained in a single(!) scattering experiment, if we ignore all other noise sources.

The inherent difference in the quantum computational complexity of permanents and determinants is therefore expressed, in our particular setup, in the *variance* of the possible outcomes of measurements. Although we have discussed a specific physical setup, we believe that this is a generic result, up to polynomial factors, in any QM many-particle experiment.

### 9.2.5 Permanent of a Unitary Hermitian Matrix

The eigenvalues of a Unitary Hermitian matrix can be either $+1$ or $-1$. In this case the decomposition in eq. (13) contained only two terms:

$$\mathrm{per}(\mathbf{U}) = P(+1) - P(-1). \tag{9.26}$$

In this case, in order to approximate the permanent we need to estimate just two probabilities, at least one of them is of $O(1)$. The accuracy of such an estimation will be of order $1/M$, were $M$ is the number of experiments. Since the value of a permanent of a unitary, Hermitian matrix can be any real value between $-1$ and $+1$, this may give a good approximation for certain matrices. However, the value of the permanent any be exponentially small, and in this case the relative approximation may not be good enough. Still, there is no way that we know of to get a better approximation.

### 9.2.6 Discussion

In this work we consider the possibility of using the symmetry properties of quantum mechanical systems of indistinguishable particles to evaluate the determinant or permanent of a given real valued Hermitian matrix. We show that it is rather easy to construct a quantum mechanical scattering experiment that yields the permanent and determinant of a matrix, as its expectation value, for bosons and fermions respectively.

While such an experiment can be prepared and performed in a polynomial time in the size of the matrix, we traced the manifestation of the notorious difference between the computational complexity of determinants and permanents to the intrinsic variance of the measurement. While there is usually an exponentially large variance in the measurements of the product observable $\Omega = \prod_{i=1}^{n} \omega_i$ in the setup we described for an $n$-boson system, there will be no variance at all for an $n$-fermion system, or the estimation of the determinant. Since this is an intrinsic QM uncertainty, we consider this result to be generic for any possible many-particle quantum mechanical computation of the permanent, but proving it remains an open problem.

## Acknowledgment

We would like to thank Nati Linial and Avi Wigderson for useful discussions.

# Part VI

# Bibliography

# Bibliography


[1] Achlioptas, D., L. Kirousis, E. Kranakis and D. Krizanc, Rigorous Results for Random $(2+p)$–SAT, *submitted to Phys. Rev. E.* A preliminary version appeared in RALCOM 97, 1997.

[2] Ackley, D. H. Hinton, G.E. and Sejnowski, T.j. A Learning Algorithm for Boltzmann Machines. *Cognitive Science* 9,pp. 147-169, 1985.

[3] Aiello, W. and Mihail, M. Learning the Fourier spectrum of probabilistic lists and trees. In *Proceedings of SODA 91, (ACM)* pp. 291-299, 1991.

[4] Alon, N., S. Ben-David, N. Cesa-Bianchi and D. Haussler, Scale-sensitive Dimensions, Uniform Convergence, and Learnability, *JACM 44*,4 pp. 616-631, 1997.

[5] Alon, N., P. Frankl, and V. Rödl, Geometrical Realization of Set Systems and Probabilistic Communication Complexity, *STOC-85*,pp. 277-280, 1985.

[6] B. Aspvall, M.F. Plass and R.E. Tarjan, A linear time algorithm for testing the truth of certain quantified Boolean formulas, *Inf. Process. Lett.***8**, 121, 1979.

[7] Bayes, T. An essay towards solving a problem in the doctrine of chances. *Phil. Trans.* 3 pp 370-418, 1763. Reproduced in *Two Papers by Bayes* ed. W.E. Deming. New-York: Hafner. 1963.

[8] Barron, A. R. Universal Approximation Bounds for Superposition of a Sigmoidal Function *IEEE Transactions on Information Theory,* Vol. 39, No. 3. pp 930-945, 1993.

[9] Baxt, W., The application of the artificial neural network to clinical decision making, In *Conference on Neural Information Processing Systems – Natural and Synthetic*, November 30- December 3, Denver, Co. 1992.

[10] Baxter, J. The Canonical Distortion Measure for Vector Quantization and Function Approximation In *Proceedings of the Fourteenth International Conference on Machine Learning,* July 1997.

[11] Baxter, J. A Bayesian/Information Theoretic Model of Learning to Learn via Multiple Task Sampling. *Machine Learning* Vol. 28, No. 1. pp 7-39. 1997.

[12] Breiman, L., J. H. Friedman, R. A. Olshen and C. J. Stone, *Classification and Regression Trees*, Belmont, CA: Wadsworth International Group, ISBN 0-534-98053-8, 0-534-98054-6 (pbk.), 1984.

[13] Ben-David, S. and A. Litman, "Combinatorial Variability of Vapnik Chervonenkis Classes", *Preprint.*

[14] Bishop, C. M., *Neural Network for Pattern Recognition*, Clarendon Press, Oxford, **ISBN** 0 19 853864 2 *(Pbk)*, 1995.

[15] Blauert, J. Modeling of interaural time and intensity difference discrimination. Psychophysical, Physiological and Behavioral Studies in Hearing, edited by G. vanden Brink and F. Bilsen. Delft, The Netherlands: Delft University Press, pp. 421-424, 1980.

[16] Blumer, A., A. Ehrenfeucht, D. Haussler, and M.K. Warmuth *Learnability and the VC dimension* Journal of the Association for Computing Machinery, 36(4), pp 929-965, 1989.

[17] Bollobás, B., Random Graphs, Academic Press, London, 1985.

[18] Boser, B., I. Guyon, and V. N. Vapnik, A training algorithm for optimal margin classifiers, *Proceedings of COLT-92, ACM, NY* pp. 144-152, 1992.

[19] BRODER, Andrei, Alan FRIEZE, and Eli UPFAL, On the satisfiability of random 3-CNF , *Proc. of Symposium on Discrete Algorithms* pp. 322–330. 1993.

[20] Broomhead, D. S. and D. Lowe, Multivariable functional interpolation and adaptive networks. *Complex systems,* 2. pp 321-355, 1988.

[21] Brualdi, R. and H. Ryser, *Combinatorial Matrix Theory, Encyclopedia of Mathematics and Its Applications* **39**, Cambridge University Press, pp. 198-235, 1991.

[22] Caruana, R., Multitask Learning, *Machine Learning* 28, no. 1. pp 41-75, 1997.

[23] Casselmann, F.L., D.F. Freemann, D.A. Kerringan, S.E. Lane, N.H. Millstrom and W.G. Nichols Jr., A neural network-based passive sonar detection and classification design with a low false alarm rate. *IEEE conference on Neural Networks for Ocean Engineering*, pp 49-55, Washington, DC. 1991.

[24] Cauchy, Augustin-Louis, Mèmoire sur les fonctions qui ne peuvent obtenir que deux valeurs ègales et de signes contraires per suite des transpositions opèrèes entre les variables qu'elles renferment, *J.Éc. Polyt.* **10** Cah. 17, pp. 29-112, 1812.

[25] Chang, E. I. and R. P. Lippmann, A boundary hunting radial basis function classifier which allocates centers constructively, *Advances in Neural Information Processing Systems* 4, ed. J. Moody, S. J. Hanson and R. P. Lippmann. (San Mateo, CA. Morgan Kaufmann) pp. 139-146, 1992.

[26] S. Chen, *Basis Pursuit.* PhD thesis, Department of Statistics, Stanford University, November 1995.

[27] Chen, S., D. Donoho and M. Saunders. Atomic decomposition by basis pursuit. Technical report, Department of Statistics, Stanford University, November 1995.

[28] Chao, Ming-Te, and J. Franco, Probabilistic analysis of two heuristics for 3-satisfiability problem, *SIAM Journal on Computing,* **15** pp. 1106–1118, 1986.

[29] CHAO, Ming-Te, and J. FRANCO, Probabilistic analysis of a generalization of the unit-clause literal selection heuristics for the $K$ satisfiable problem, *Information science* **51** 289–314, 1990.

[30] Chvátal, Vašek and B. Reed, Mick gets some (the odds are on his side), *Proc. 33rd IEEE symposium on Foundation of Computer Science* 620–627, 1992.

[31] Chvátal and Endre Szemerédi, Many hard examples for resolution, *Journal of the ACM* **35** pp. 759–768, 1988.

[32] Cohen, W. W. The Dual DFA Learning Problem: Hardness Results for Programming by Demonstration and Learning First-Order Representations. *Proceedings of COLT-96, ACM, NY,* pp. 29-40, 1996.

[33] S.A. Cook, The complexity of theorem–proving procedures, in *Proc. 3rd Ann. ACM Symp. on Theory of Computing*, Assoc. Comput. Mach., New York, 151, 1971.

[34] Cover, T. M. Geometrical and Statistical Properties of Linear Inequalities with Application in Pattern Recognition. *IEEE Trans. Electronic Computers* 14, pp 326-334. 1965.

[35] Crawford,J. M. and L. D. Auton, Experimental Results on the Crossover Point in Random 3-SAT, *Artificial Intelligence* 81, 1-2, pp. 31-57, 1996.

[36] Crisanti, A., and H. Sompolinsky, Dynamics of spin systems with randomly asymmetric bonds: Langevin dynamics and the spherical model. *Phys. Rev. A* 36, pp. 4922-4939, 1987.

[37] Davis, M., and Putnam, H., A computing procedure for quantification theory, *J. Assoc. Comput. Mach.*, **7** ,pp. 201-215, 1960.

[38] Dayan, P. Hinton, G. E., Neal, R. M, and Zemel, R. S. The Helmholtz machine, *Neural Computation* 7, 889-904. 1995.

[39] Devroye, L. Györfi, L. and Lugosi, G.. *A Probabilistic Theory of Pattern Recognition.* Springer, **ISBN** 0-387-94618-7. 1997.

[40] Dubois, O. Counting the number of solutions for instances of satisfiability, *Theoretical Computing Science* **81** pp. 49–64, 1991.

[41] Dubois, O., and Y. Boufkhad, A general upper bound for the satisfiability threshold of random K–SAT formulas, *Journal of Algorithms 24*, pp. 395-420, 1997.

[42] Dubois, O. and Carlier, J., Probabilistic approach to satisfiability problem, *Theoretical Computing Science* **81** pp. 65–75, 1991.

[43] Duda, R. O. and Hart, P.E. *Pattern Classification and Scene Analysis*, JOHN WILEY & SONS, NY, ISBN 0 471 22361 1, 1973.

[44] Duda, R. O. and Hart, P.E. *Pattern Classification and Scene Analysis* pp 134-138, JOHN WILEY & SONS, NY, 1973.

[45] Dudley, R. M. Universal Donscer classes and metric entropy, *Ann. Prob.* 15, (4), pp 1306-1326, 1987.

[46] EDWARDS, S. F. and Phil W. ANDERSON, Theory of spin glasses, *J. Phys. F* **5**, pp. 965-974, 1975.

[47] Field, D. J. What is the goal of sensory coding?, *Neural Computation* 6, pp. 559-601, 1994.

[48] Fisher, R. A. The use of multiple measurements in taxonomic problems, *Ann. Eugenics,* 7, part II, 179-188, 1936. also in *Contributions to Mathematical Statistics* John Wiley, New-York, 1952.

[49] Fisher, R. A. *Contributions to Mathematical Statistics* John Wiley, New-York, 1952.

[50] Floyd, S. and M. Warmuth, Sample compression, learnability, and the Vapnik-Chervonenkis dimension, *Machine Learning*,21:1-36, 1995

[51] Franco, J. and M. Paull, Probabilistic analysis of the Davis Putman procedure for solving the satisfiability problem", *Discrete Applied Mathematics*, **5** pp. 77-87, 1983.

[52] Freund, Y. Boosting a weak learning algorithm by majority In *Proceedings of the Third Workshop on Computational Learning Theory,* San Mateo, CA. Morgan Kaufmann. pp 202-216, 1990.

[53] Freund, Y. *Data Filtering and Distribution Modeling Algorithms for Machine Learning.* Ph. D. thesis, University of California at Santa Cruz, 1993. Retrievable from: ftp.cse.ucsc.edu/pub/tr/ucsc-crl-93-37.ps.Z

[54] Freund, Y. and Schapire, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the Second European Conference on Computational Learning Theory,* 1995.

[55] Freund, Y, H. S. Seung, E. Shamir and N. Tishby. Selective Sampling Using The Query By Committee Algorithm. *Machine Learning* 1996.

[56] Friedgut, E., Necessary and sufficient conditions for sharp thresholds, preprint, presented at DIMACS 1997.

[57] Frieze, A., and S. Suen, Analysis of two simple heuristics on a random instance of K–SAT, *Journal of Algorithms 20*, pp. 312-335, 1996.

[58] Fu, Y.-T. and P.W. Anderson, *Lectures in the Science of Complexity*, D. Stein (ed.) p. 815, Addison-Wesley, 1989.

[59] Gallant, S., *Neural Network Learning and Expert Systems*, ISBN 0-262-07145-2, MIT Press, Cambridge, MA. 1993.

[60] Gallant, S. I. and Smith, D. Random Cells: An Idea Whose Time Has Come and Gone... and Come Again? *IEEE International Conference on Neural Networks,* San Diego, CA, vol. II, pp 671-678, 1987.

[61] M. Garey and D.S. Johnson, *Computers and Intractability; A guide to the theory of NP–completeness*, W.H. Freeman and Co., San Francisco, 1979;

[62] Girosi, F. An equivalence between sparse approximation and support vector machines. Technical report. A.I. Memo No. 1606. C.B.C.L. paper no. 147, M.I.T. 1997.

[63] Girosi, F. M. Jones and T. Poggio, Regularization Theory and neural networks architectures. *Neural Computation,* 7. pp. 219-269. 1995.

[64] Gold, E. M. Language identification in the limit, *Information and Control,* Vol. 10, pp 675-695, 1967.

[65] Guyon, I., *Neural Networks and Applications* Computer Physics Reports, Amsterdam: Elsevier.

[66] Györgyi, G, and N. Tishby, Statistical theory of learning a rule. In K. Thuemann and R. Koeberle, editors, *Neural Networks and Spin Glasses.* World Scientific, 1990.

[67] Geman, S. and Geman, D. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-6, pp. 721-741, 1984.

[68] Gibbs, M. N. and D. J. C. MacKay Variational Gaussian Process Classifiers. preprint, Cavendish Laboratory, Cambridge, UK. 1997.

[69] Harrison, R., S. Marshall and R. Kennedy, The early diagnosis of heart attacks: A neurocomputational approach, *International Joint Conference on Neural Networks,* Vol. 1, pp 1-5, Seattle, WA., 1991.

[70] Haussler, D., Decision-theoretic generalization of the PAC model for neural net and other learning applications, *Information and Computation,* 95(2): 129-161, 1991.

[71] Haussler. D, M. Kearns, and R.E. Schapire, Bounds on the sample complexity of Bayesian learning using information theory and the VC dimension, *Machine Learning,* 14:83-113, 1994.

[72] Haussler D., M. Kearns, H. S. Seung, and N. Tishby, Rigorous Learning Curve Bounds from Statistical Mechanics, *Proceedings of COLT-94, ACM, NY,* pp.76-87, 1994.

[73] B. Hayes, *American Scientist* **85**, 108, 1996.

[74] Hebb, D.O. *The Organization of Behavior: A Neuropsychological Theory.* New York: Wiley, 1949.

[75] J.A. Hertz, A. Krogh, R.G. Palmer, *Introduction to the theory of neural computation*, Addison-Wesley, Redwood City (CA), 1991.

[76] N. Intrator and S. Edelman, Learning low-dimensional representations via the usage of multiple-class labels. *Network: Computation in Neural Systems,* vol. 8, no. 3, pp. 259-281, 1997.

[77] Ivanov, V. V. On linear problems which are not well posed. *Soviet Math. Docl.,* 3, (4), pp. 981-983. 1962.

[78] Jacobs, D.J., and M. F. Thorpe, *Phys. Rev. Lett.* **75**, 4051, 1995.

[79] Jonsson, D. , Asymptotic Distributions of Functions of the Eigenvalues of Some Random Matrices for Nonnormal Populations, *Jornal of multivariate analysis*, 12, pp. 39-63, 1982.

[80] Jordan, M.I. and R. A. Jacobs. Learning to control an unstable system with forward modeling, In *Advances in Neural Information Processing Systems 2* (D.S. Touretzky, ed), pp 324-331. San Mateo, CA: Morgan Kaufmann. 1990.

[81] Kadirkamanathan, V., M. Niranjan and F. Fallside: sequential adaptation of radial basis function neural networks and its application to time-series analysis, in *Advances in Neural Information Processing Systems 3* ed. R. P. Lippmann, J. Moody and D. S. Touretzky, San Mateo, CA: Morgan Kaufmann. pp. 721-727, 1991.

[82] Kamath, A., R. Motwani, K. Palem, and P. Spirakis, Tail bounds for occupancy and the satisfiability thresholds conjecture, *Proc. 35rd IEEE symposium on Foundation of Computer Science* pp. 592–603, 1994.

[83] Kearns, M. and Y. Mansour, On the Boosting Ability of Top-Down Decision Tree Learning Algorithms, *preprint,* Invited to the special issue of JCSS.

[84] Kearns, M. and L. G. Valiant, Cryptographic limitations on learning Boolean formulae and finite automata, *Proceedings of the twenty-first annual ACM symposium on theory of computing,* pp. 433-444, 1989.

[85] Kearns, M. J. and U. V. Vazirani *An Introduction to Computational Learning Theory* MIT press, Cambridge, MA. 1994.

[86] Kirkpatrick, S., G. Györgyi, N. Tishby, and L. Troyansky, The Statistical Mechanics of K-Satisfaction, *Advances in Neural Information Processing Systems* **6**, 439-446, 1993.

[87] S. Kirkpatrick, C.D. Gelatt Jr. and M.P. Vecchi, Optimization by simulated annealing, *Science* **220**, pp. 671-680, 1983.

[88] Kirkpatrick, S. and B. Selman, Critical behavior in the satisfiability of random Boolean expressions, *Science* **264** pp. 1297-1301, 1994.

[89] Kirkpatrick, S. and G. Toulouse, Configuration space analysis of traveling salesman problems, *Journal de Physique* **46**, pp. 1277-1292, 1985.

[90] Kirousis, L., E. Kranakis and D. Krizanc, Approximating the unsatisfiability threshold of random formulas, *Proceedings of the 4th European Symposium on Algorithms*, pp. 27-38, 1992.

[91] Krogh, A. and Vedelsby, J., Neural network ensembles, cross-validation and active learning, In *Advances in Neural Information Proceeding Systems 7,* eds. G. Tesauro, D. S. Touretzky, and T. Leen, MIT Press, Cambridge, Massachusetts, 1995.

[92] Kushilevitz, E. and Mansour, Y. Learning decision trees using the Fourier spectrum. In Proceedings of the $23^{rd}$ Annual ACM symposium on Theory of Computing, pp. 455-464, 1991 and In *SIAM Journal on Computing* 22 (6), pp. 1331-1348, 1993.

[93] LeCun, Y, .B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard and L.D. Jackel, Handwritten digit recognition with a back-Propagation network. In *Advances in Neural Information Processing Systems 2* (D.S. Touretzky, ed), pp 598-605. San Mateo, CA: Morgan Kaufmann, 1990.

[94] Lee, Y. Handwritten digit recognition using K nearest-neighbors, radial basis function and backpropagation neural networks, *Neural Computation* 3, pp 440-449, 1991.

[95] Lee, Y. and R. P. Lippmann, Practical characteristics of neural network and conventional pattern classifiers on artificial and speech problems, in *Advances in Neural Information Processing Systems 2* ed. D. S. Touretzky, San Mateo, CA: Morgan Kaufmann. pp. 168-177, 1990.

[96] Light, W. A., Some aspects on radial basis function approximation. *Approximation Theory, Spline Functions and Applications* ed. D. S. Singh, NATO ASI series, 256 (Dordrecht: Kluwer) pp 163-190, 1992.

[97] N. Linial, Y. Mansour and N. Nisan, Constant depth circuits, Fourier transform and learnability. In $30^{th}$ *Annual ACM Symposium on Theory of Computer Science, Research Triangle Park, NC.* pp 574-579, 1989 and in: *JACM* 40, (3) pp. 607-620, 1993.

[98] Littlestone, N., and M. Warmuth, Relating Data Compression and Learnability, *Unpublished manuscript*, 1996.

[99] Littlestone, N., and M. Warmuth, The weighted Majority Algorithm. Technical Report UCSC-CRL-89-16, Computer Research Laboratory, University of Santa Cruz, July, 1989.

[100] Lowe, D. Radial-Basis Function Networks, In M. A. Arbib (Ed.) *The Handbook of Brain Theory and Neural Networks.* Cambridge, MA: MIT Press.

[101] Lowe, D. and A. R. Webb, Exploiting prior knowledge in network optimization: An illustration from medical prognosis. *Network,* 1. pp. 299-323, 1990.

[102] MacKay, D. J. C. Probable networks and plausible predictions – A review of practical Bayesian methods for supervised neural networks. *Network: Computation in Neural Systems* 6. pp. 469-505. 1995.

[103] Mansour, Y. and Sahar, S. Implementation Issues in the Fourier Transform Algorithm, In: *Advances in Neural Information Processing Systems, 8* pp. 260-266, 1995.

[104] May B.J. and A. Y., Huang Spectral cues for sound localization in cats: A model for discharge rate representations in the auditory nerve. *J. Acoust. Soc. Am.*, 101 pp. 2705-2719, 1997.

[105] McCulloch, W. S. and W. H. Pitts, A logical calculus of the ideas immanent in nervous activity, *Bulletin of Math. Biophysics,* 5. pp 115-133. 1943

[106] M. Mézard, G. Parisi, M.A. Virasoro *Spin Glass Theory and Beyond*, World Scientific, Singapore, 1987.

[107] Middlebrooks, C.J. Clock, A.E., Xu,L. and Green, D.M.: A Panoramic Code for Sound Location by Cortical Neurons. *Science*, vol. 264: 842:844, 1994.

[108] Minc, Henrik, *Permanents, Encyclopedia of Mathematics and Its Applications* **6**, Addison-Wesley Publishing Company, Reading, Mass. 1978.

[109] Monasson, Remi and Riccardo Zecchina, The entropy of the K-satisfaction problem, *Phys. Rev. Lett.* **76** pp. 3881-3884, 1996.

[110] R. Monasson and R. Zecchina, *Statistical Mechanics of the Random K-Satisfiability Problem*, *Phys. Rev.* **E 56**, 1357, 1997.

[111] R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman , L. Troyansky, Phase Transition and Search Cost in the $2+p$ SAT Problem, proceedings of *PhysComp96*, T. Toffoli, M. Biafore, J. Leão eds., Boston, 1996.

[112] R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman , L. Troyansky, Typical-Case Complexity Results From a New Type of Phase Transition. Submitted to *Nature*, December 1998.

[113] R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman , L. Troyansky, 2+p–SAT: Relation of Typical-Case Complexity to the Nature of the Phase Transition, Submitted to *Random structures and algorithms*, December 1998.

[114] Moody, J. E. and C. J. Darken. Fast learning in neural network of locally-tuned processing units. *Neural Computation,* 1. pp. 281-294, 1989.

[115] Moukarzel, C. and P. M. Duxbury, *Phys. Rev. Lett.* **75**, p. 4055. 1995.

[116] Musavi, M. T., W. Ahmed, K. H. Chan, K. B. Faris and D. M. Hummels, On the training of radial basis function classifiers, *Neural Networks* 5, pp 595-603, 1992.

[117] Musicant, A.D. and R.A. Butler. Influence of monaural spectral cues on binaural localization, J. Acoust. Soc. Am. 77:202-208, 1985.

[118] Nadir, K. J-M. Vesin and L. Eyer, Spectral estimation of inequally-spaced data via radial basis function interpolation, *XIVth Colloque sur le traitement du signal et ses applications* (GRETSI) (Juan-Les-Pins) pp. 217-220, 1993.

[119] Nandy, D. and J. Ben-Arie. An Auditory Localization Model Based on High-Frequency Spectral Cues, Biomedical Engineering, Vol. 24. pp. 621-638, 1996.

[120] arendra, K.S. and K. Parthasarathy, Identification and control of dynamical systems using neural networks *IEEE Transactions on Neural Networks* **1** 4-27, 1990.

[121] Neal, R. M. Connectionist learning of belief networks, *Artificial Intelligence* 56, 71-113. 1992.

[122] Neal, R. M. Bayesian learning for neural networks, Ph.D. dissertation, Dept. of computer science, University of Toronto. 1995.

[123] Nelken, I. and Young, D., Two Separate Inhibitory Mechanisms Shape the Response of Dorsal Cochlear Nucleus Type IV Units to Narrowband and Wideband Stimuli. *Journal of Neurophysiology,* vol. 71: 2446-2462, 1994.

[124] Neti, C., E. D. Young and M. H. Schneider. Neural network models of sound localization based on directional filtering by the pinnae. *J. Acoust. Soc. Am.* 92, pp. 3140-3156, 1992.

[125] Ng, K. and R. P. Lippmann, Practical characteristics of neural network and conventional pattern classifiers. In *Advances in Neural Information Processing Systems 3,* (R. P. Lippmann, J. E. Moody and D. S. Touretzky, eds.) pp. 970-976, 1991.

[126] Nguyen, D. and B. Widrow, The truck backer-upper: An example of self-learning in neural networks, *International Joint Conference on Neural Networks,* Vol. 2, pp 357-363, Washington, DC. 1989.

[127] Opitz, D. and J. Shavlik, Generating accurate and diverse members of neural-network ensemble. In: *Advances in Neural Information Processing Systems, 8* D. Touretzky, M. Mozer and M. Hasselmo, eds. MIT press, Cambridge, MA. 1995.

[128] Osuna, E. Freund, R. and Girosi, F. An Improved Training Algorithm for Support Vector Machines. In *Proc. of IEEE NNSP 1997, Amelia Island, FL,* 1997.

[129] Pach, J. and P. Agarwal, *Combinatorial Geometry*, J. Wiley, New York, (1995).

[130] Papadimitriou, C. H., On selecting a satisfyinging truth assignment. *Proc. of the 32nd symposium on Foundation of Computer Science.*, pp. 163-169, 1991.

[131] Pauli, W., The connection between spin and statistics, *Phys. Rev.***58**, pp. 716-722, 1940.

[132] Pavlov, I. P. *Conditioned Reflex: An Investigation of the Physiological Activity of the Cerebral Cortex.* G. V. Anrep (trans.) Oxford University Press, London, 1927.

[133] Persky, N. , I. Kanter and S.Solomon, Cluster Dynamics for Randomly Frustrated Systems with Finite Connectivity, *Phys. Rev. E.* 53, p. 1212, 1996.

[134] Phillips, D. Z. A technique for numerical solution of certain integral equation of the first kind. *J. Assoc. Comput. Mach.* 9, pp. 84-96. 1962.

[135] Poggio, T. and S. Edelman, A network that learn to recognize three-dimensional objects. *Nature* 343, pp 263-266, 1990.

[136] Poggio, T and F. Girosi, Networks for approximation and and learning. *Proc. IEEE* 78, pp 1481-1497, 1990.

[137] Pollard, D. *Convergence of stochastic processes,* Springer, New York. 1984.

[138] Pitt, L. and M. Warmuth, Prediction Preserving reducibility, *Journal of Computer and System Sciences*,41:430-467, 1990.

[139] Pitt, L. and L. Valiant, Computational limitations on learning from examples. *Journal of the ACM*, 35:965-984, 1988.

[140] Powell, M. J. D. Radial Basis Functions for Multi-variables Interpolation: A Review. IMA conference on Algorithms for Approximation of Functions and Data, RMCS, Shrivenham, England. 1985.

[141] Pratt, L. and B. Jennings, 'A Survey of Transfer Between Connectionist Networks, *Connection Science*, 8, No. 2. pp. 163-184, 1996.

[142] Quinlan, J. R., *C4.5: Programs for Machine Learning.* San Mateo, CA: Morgan Kaufmann. 1993.

[143] Rice, J.J., May, B.J., Spirou, G.A., and Young, E.D.: Pinnae Based Spectral Cues for Sound Localization in Cats, *Hearing Res.*, vol. 58: 132-152, 1992.

[144] E. Rich and K. Knight, *Artificial Intelligence,* second edition, pp. 447-484. McGraw-Hill inc. ISBN 0-07-052263-5, 1991.

[145] Rissanen, J. Modeling by short data description, *Automatica*, 14, pp. 465-471, 1978.

[146] Rosenblatt, F. The perceptron – a perceiving and recognizing automaton, Report 85-460-1, Cornell Aeronautical Laboratory, Ithaca, New-York, January 1957.

[147] Royall, R. *A Class of Nonparametric Estimators of a Smooth Regression Function.* PhD Thesis, Stanford University, Stanford, CA. 1966.

[148] Rumelhart, D. E., G. E. Hinton and R. J. Williams. Learning representations by back-propagating errors. *Nature* (London), 323, pp. 533-536. 1986.

[149] Rosenblatt, F., *Principles of Neurodynamics*, Spartan, New-York, 1962.

[150] Saha, A. and J. D. Keeler, Algorithms for better representation and faster learning in radial basis function networks in *Advances in Neural Information Processing Systems 2* ed. D. S. Touretzky, San Mateo, CA: Morgan Kaufmann. pp. 482-489, 1990.

[151] Saha, A., J. Christian, D. S. Tang and C.-L. Wu, Oriented non-radial basis functions for image coding and analysis. In *Advances in Neural Information Processing Systems 3*, (R. P. Lippmann, J. E. Moody and D. S. Touretzky, eds.) pp. 728-734 1991.

[152] Sanger, T. D. Optimal unsupervised learning in a single-layer linear feedforward neural network, *Neural Networks*, vol. 12. pp. 459-473, 1989.

[153] Sanner, R. M. and J-J. E. Slotine, Gaussian networks for direct adaptive control, *IEEE Trans. Neural Networks* 3. pp. 482-489, 1990.

[154] Schapire, R. E., The strength of weak learnability, *Machine Learning,* 5 (2), pp 197-227, 1990.

[155] Schapire, R.E., and Y. Freund, A decision theoretic generalization of on-line learning and its application to boosting, In *EuroColt '95*, pp. 23-27, Springer Verlag, 1995.

[156] Schapire, R.E., Y. Freund and P. Bartlett, Boosting the margin: A new explanation for the effectiveness of voting methods, *Machine Learning: Proceedings of the 14th International Conference*, Nashville, TN., 1997.

[157] Schneider, J., C. Froschhammer, I. Morgenstern, T. Husslein, and J. M. Singer, Searching for Backbones – an efficient parallel algorithm for the travelling salesman problem, *Computer Physics Communications 96* pp. 173-188, 1996.

[158] ejnowski, T.J. and C. R. Rosenberg, Parallel networks that learn to pronounce English text, *Complex Systems* **1**, pp 145-168, 1987.

[159] Selman, B. and S. Kirkpatrick, Critical behavior in the computational cost of satisfiability testing, *Artificial Intelligence* **81**, 273, 1996.

[160] B. Selman and H. Kautz, Domain independent version of GSAT solving large structured satisfiability problems, *IJCAI*, 1993.

[161] Selman, B., H. Kautz, and B. Cohen, Local search strategies for satisfiability testing, In *Cliques, Coloring and Satisfiability: Second DIMACS Implementation Challenge*, American Mathematical Society, 1996.

[162] Shepard, R. N. Towards a universal law of generalization for psychological science. *Science*, 237:1317-1323, 1987.

[163] Shor, P., Algorithms for Quantum Computations: Discrete Logarithms and Factoring, *Proceedings of the 35th Annual Symposium on the Foundation of Computer Science*, (Goldwasser Shafi ed.), IEEE Computer Society Press, Los Alamitos, CA., pp. 124-134, 1994.

[164] Simon, D., On the Power of Quantum Computation, *Proceedings of the 35th Annual Symposium on the Foundation of Computer Science*, (Goldwasser Shafi ed.), IEEE Computer Society Press, Los Alamitos, CA.,pp. 116-123, 1994.

[165] Smolensky, P. Information Processing in Dynamical Systems: Foundations of Harmony Theory. in Rumelhart, D.E. and McClelland, J.L. (eds.) *Parallel Distributed Processing: Explorations in the microstructure of Cognition,* vol. 1. 194-281. MIT Press, Cambridge, 1986.

[166] Sollich, P. and A. Krogh, Learning with ensembles; How over-fitting can be useful. In *Advances in Neural Information Proceeding Systems 8*, MIT Press, Cambridge, Massachusetts, 1996.

[167] H. Sompolinsky, N. Tishby and H. Seung. Learning from examples in large neural networks. *Phys. Rev. Lett.*, 65:1683-1686, 1990.

[168] Anglès d'Auriac, J.-C. and Sourlas, N., The $3d$ random field Ising model at zero temperature, Europhysics Lett. 39, 473-478, pp. 1997.

[169] N. Tishby, E. Levin and S. Solla. Consistent inference of probabilities in layered networks: prediction and generalization. In *IJCNN International joint Conference on Neural Networks,* volume II, pp. 403-409. IEEE, 1989.

[170] Tikhonov, A. N., On solving ill-posed problems and method of regularization, *Doklady Akademii Nauk USSR,* 153, pp. 501-504. 1963.

[171] N. Tishby, private communication.

[172] L. Troyansky and Tishby, N., Moments of satisfaction: statistical properties of a large random K-CNF formulae. *PhysComp96: Proceedings of the fourth workshop on physics and computation,* T. Toffoli, M. Biafore and J. Leão eds., pp. 308-313, New England Complex Systems Institute, 1996.

[173] L. Troyansky and Tishby, N., Permanent Uncertainty: On the quantum evaluation of the determinant and the permanent of a matrix, *PhysComp96: Proceedings of the fourth workshop on physics and computation,* T. Toffoli, M. Biafore and J. Leão eds., pp. 314-318, New England Complex Systems Institute, 1996.

[174] Valiant, L. G., The Complexity of Computing the Permanent, *Theoretical Computer Science* **8**, pp. 189-201, 1979.

[175] Valiant, L. G. A theory of the learnable, *Communications of the ACM,* Vol. 27, No. 11, pp. 1134-1142, 1984.

[176] Vapnik82 Vapnik, V., Estimation of Dependences Based on Empirical Data. Springer-Verlag, New-York, 1982.

[177] Vapnik, V., *The Nature of Statistical Learning Theory* pp. 119-156, Springer-Verlag, New-York, 1995.

[178] Vapnik, V. N. and Chervonenkis, A. Ya. On The uniform convergence of relative frequencies of events to their probabilities, *Theor. Prob. Appl.*, 16 264-280, 1971.

[179] Vapnik, V. N., S. E. Golowich and A. Smola. Support vector methods for function approximation. In *Advances in Neural Information Processing Systems 9,* San-Mateo, CA. Morgan Kaufmann Publishers. pp. 281-287. 1997.

[180] Wachter, K. W., The strong limits of random matrix spectra for sample matrices of independent elements. *Ann. Probb.* 6, pp. 1-18, 1978.

[181] Wahba, G. Smoothing and ill-posed problems. In *Solution methods for integral equations and applications,* (editor: M. Goldberg) pp. 183-194. Plenum Press, New-York, 1979.

[182] Wahba, G. Splines Models for Observational Data. *Series in Applied Mathematics, Vol. 59, SIAM, Philadelphia,* 1990.

[183] Wigner, E. P. On the distributions of the roots of certain symmetric matrices. *Ann. of Math.* 67, pp. 325-327, 1958.

[184] Williams, C. K. I. and Rasmussen, C. E. Gaussian Processes for regression. In: Advances in Neural Information Processing Systems 8, ed. by D. S. Touretzky, M. C. Mozer and M. E. Hasselmo. pp. 514-520, 1996.

[185] E. D. Young, private communication.

[186] Young, E.D., Shofner, W.P., White, J.A., Robert, J.M. and Voigt, H.F.: Response Properties of Cochlear Nucleus Neurons in Relationship to Physiological Mechanisms. in *Auditory Function: Neurobiological Bases of Hearing* ed: Edelman, G.M, Gall, W.E. and Cowan, W.M. Jhon Wiley & Sons, NY, 1988.

[187] Zhu, H., C. K. I. Williams, R. Rohwer and M. Morciniec, Gaussian Regression and Optimal Finite Dimensional Linear Models. Technical Report NCRG/97/011, Dept. of Computer Science and Applied Mathematics, Aston University, Birmingham, UK. 1997.

[188] Zuckerman, D. NP-Complete Problems Have a Version That's Hard to Approximate, *Proceedings of the 8th Annual Conference on Structure in Complexity Theory.* IEEE Computer Society Press, Los Alamitos, CA, pp. 305-123, 1993.

[189] Zwislocki, J. and R. Feldman, Just noticeable differences in dichotic phase. *J. Acoust. Soc. Am.* 28 pp. 860-864, 1956.