Robust Trainability of Single Neurons

Klaus-U. Höffgen, Hans-U. Simon*
Lehrstuhl Informatik II
Universität Dortmund
D-4600 Dortmund 50
hoeffgen,simon@nereus.informatik.uni-dortmund.de

Kevin S. Van Horn
Computer Science Department
Brigham Young University
Provo, UT 84602
kevin@bert.cs.byu.edu

June 3, 1994

*These authors gratefully acknowledge the support of Bundesministerium für Forschung und Technologie grant 01IN102C/2. The authors take responsibility for the content.

Abstract

It is well known that (McCulloch-Pitts) neurons are efficiently trainable to learn an unknown halfspace from examples, using linear-programming methods. We want to analyze how the learning performance degrades when the representational power of the neuron is overstrained, i.e., if more complex concepts than just halfspaces are allowed. We show that the problem of learning a probably almost optimal weight vector for a neuron is so difficult that the minimum error cannot even be approximated to within a constant factor in polynomial time (unless RP = NP); we obtain the same hardness result for several variants of this problem. We considerably strengthen these negative results for neurons with binary weights 0 or 1. We also show that neither heuristical learning nor learning by sigmoidal neurons with a constant reject rate is efficiently possible (unless RP = NP).

To Appear in *Journal of Computer and System Sciences*

1 Introduction

A (McCulloch-Pitts) neuron computes a linear threshold function

$$f(x) = \begin{cases} 1 & \text{if } w_1x_1 + \ldots + w_nx_n \geq \tau, \\ 0 & \text{if } w_1x_1 + \ldots + w_nx_n < \tau. \end{cases}$$

In order to learn a target concept c (an unknown subset of the Euclidean n-space $\mathcal{R}^n$) with respect to distribution D (an unknown probability distribution on $\mathcal{R}^n$), we want to apply a learning algorithm L to a sample of labeled and randomly chosen training examples. The objective of L is to adjust the weights w_i and τ of the neuron to the concept at hand. A perfect adjustment with $x \in c \iff f(x) = 1$ exists iff the concept is a linear halfspace. If we restrict the class of concepts to halfspaces, we may use linear programming for finding an adjustment which is consistent to the sample. This leads to the well-known [7] fact that halfspaces are PAC learnable (and neurons are trainable to PAC-learn them). This positive result is valid for variable dimension n. Throughout the paper, we will demand that our learning algorithms work properly for variable n.

We want to analyze how learning performance degrades when the representational power of the neuron is overstrained, i.e., if more complex concepts than just halfspaces are allowed. A rigorous analysis of this question is based on the notion of PAO (probably almost optimal) learnability, because we must not insist on probably almost correct hypotheses. If PAO learning is possible for arbitrary concepts, we speak of robust learning and robust trainability. We will show that a neuron is not robustly trainable—i.e., it is not in general possible to find a probably almost optimal weight vector in polynomial time—nor is it even possible to approximate optimality to within a constant factor in polynomial time, unless RP = NP. As a corollary we find the same hardness result for learning monomials and 1-DL formulas. We considerably strengthen these negative results for neurons with binary weights 0 or 1. Other variants of learning more complex concepts than halfspaces by single neurons are also investigated. We show that neither heuristical learning nor learning by sigmoidal neurons with a constant reject rate is efficiently possible (unless RP = NP).

All our proofs are based on a combinatorial method that associates an optimization problem with each learning problem. If the optimization problem is NP-hard, then its corresponding learning problem is also intractable unless RP = NP.

The paper is structured as follows: Section 2 presents the formal definitions of PAO and robust learnability. In addition, it describes the aforementioned combinatorial method. Section 3 states the negative results concerning robust trainability of neurons. Section 4 proves the difficulty of even approximating optimality to within a constant factor. Section 5 presents strong negative results for the case that the weights w_i are restricted to be 0 or 1. Section 6 is devoted to heuristical learnability and learnability with constant reject rates. It is shown

that the former kind of learning is impossible for McCulloch-Pitts neurons, and the latter is impossible for sigmoidal neurons (unless RP = NP). Section 7 draws some conclusions, discusses relations between our results and other papers and mentions some open problems.

2 Robust Learning and Minimization

Let X be a set augmented with a σ-algebra $\mathcal{A}$ of events. For instance, $\mathcal{A}$ might be the powerset of X if X is countable, or the system of Borel sets if $X = \mathcal{R}^n$. D denotes an unknown, but fixed, probability measure on $\mathcal{A}$. We then define the following:

- A *concept* c on X is an element of $\mathcal{A}$.
- A *concept class* is a collection $C \subseteq \mathcal{A}$ of concepts.
- Any $x \in c$ is called a *positive example* for c, and any $x \in X \setminus c$ is called a *negative example* for c. An arbitrary element of X we simply call an *example*.
- A positive or negative example augmented with the corresponding label '+' or '−' is said to be *labeled* according to c.
- A *sample* of size m for concept c is a sequence of m examples, drawn randomly according to D^m (m independent drawings from D) and labeled according to c.

In a typical learning task, the concept is unknown to the learner but can be explored empirically by gathering more and more labeled random examples. After a while, the learner should be able to formulate a 'hypothesis' representing its guess for the unknown concept. Formally, we define the following:

- A *hypothesis class* H is another collection $H \subseteq \mathcal{A}$ of subsets of X. H represents the collection of hypotheses taken into consideration by the learner.
- We say that hypothesis $h \in H$ is ϵ-*accurate* for concept c if

$$D(h\Delta c) \leq \epsilon, \text{ where } h\Delta c = (h \setminus c) \cup (c \setminus h),$$

  i.e., if the prediction error of h is at most ϵ.
- We say that h is ϵ-*optimal* with respect to H for c if

$$D(h\Delta c) \leq \mathrm{opt}(H) + \epsilon, \text{ where } \mathrm{opt}(H) = \inf_{h \in H} D(h\Delta c),$$

  i.e. if the prediction error of h comes within ϵ of the minimum achievable by hypotheses from H.

Let $c \in C$ and distribution D on X be (arbitrarily) fixed.

- A *learning algorithm* for concept class C and hypothesis class H is a polynomial-time algorithm whose input consists of two parameters $0 < \epsilon, \delta < 1$ (*accuracy* and *confidence parameter*) and a sample of $m = m(\epsilon, \delta)$ examples (drawn randomly according to D^m and labeled according to c), and whose output is (a representation of) some $h \in H$. We demand that m be polynomially bounded in $1/\epsilon, 1/\delta$.

- A *PAO* learning algorithm L must also satisfy the following: With a probability of at least $1 - \delta$, L must output (a representation of) an ϵ-optimal hypothesis $h \in H$ for c. We demand that L satisfy this condition for all possible choices of c, D, ϵ and δ.

- We say that C is *PAO learnable* by H if there exists a PAO learning algorithm for C and H.

The above definition is modified slightly if the learning problem is parameterized by some complexity parameter n. In this case $X = (X_n)_{n \geq 1}, C = (C_n)_{n \geq 1}, H = (H_n)_{n \geq 1}$, n is part of the input, and sample size and running time of the algorithm must be polynomially bounded in n (and the other parameters). The remaining definitions of this section are also stated without the complexity parameter, but are easily generalized to the parameterized case.

An important special case of PAO learning occurs when $C \subseteq H$. In this case $c \in H$, so $\mathrm{opt}(H) = 0$ and ϵ-optimality becomes synonymous with ϵ-accuracy. Also the notion of PAO learnability becomes equivalent to the traditional notion of PAC learnability in Valiant's learning model. An extreme of the special case $C \subseteq H$ is $H = C$. In this case we follow the traditional terminology:

- We say C is *learnable* if C is PAC learnable by C.

The notion of PAO learnability becomes more interesting if $H \subset C$. An extreme situation occurs if we allow arbitrary concepts, i.e., $C = \mathcal{A}$. Let us imagine that $(c_i, D_i), i \geq 1$, is a sequence of concepts and distributions such that the corresponding values of $\mathrm{opt}(H)$ slightly increase from small values to large ones. In other words, we continously overstrain the representational power of H. It is practically important that learning algorithms be robust in the sense that their performance degrades 'gracefully' when overstraining occurs. These considerations motivate the following definition:

- We say that H *allows robust learning* if $\mathcal{A}$ is PAO learnable by H.

A combinatorial problem closely related to the learnability of a concept class C is the so-called *consistency problem* for C:

- **Consistency problem for C.**
  Input: A multiset $S = S_+ \cup S_-$ (disjoint union) of positive and negative

examples.
Question: Does there exist a concept $c \in C$ *consistent* with S, i.e., $\exists c \in C : c \cap S = S_+$?

The following result is shown in [17, 23]:

Theorem 2.1 *If RP $\neq$ NP and the consistency problem for C is NP-hard, then C is not learnable.*

Similarly, it turns out that the question whether a hypothesis class H allows robust learning is related to the so-called *Minimizing Disagreement problem*:

- **MinDis**(H).
  Input: A multiset $S = S_+ \cup S_-$ (disjoint union) of positive and negative examples, and an integer bound $k \geq 0$.
  Question: Does there exist a hypothesis $h \in H$ with at most k misclassifications on S? (If h misclassifies x, each occurrence of x in S counts as a separate misclassification.)

- We call $S = S_+ \cup S_-$ the *input sample* of MinDis(H), or simply (but somewhat ambigously) the sample.

- If C is a concept class, we say that S is *C-legal* if C contains a concept c which is consistent with S.

The following result has been already observed in [2] (although the authors state it in a technically slightly different setting.)

Theorem 2.2 *If RP $\neq$ NP and MinDis(H) restricted to C-legal input samples is NP-hard, then C is not PAO learnable by H.*

Proof. We show that a PAO learning algorithm L for C, H can be converted into a Monte Carlo algorithm A for MinDis(H). Let $S = S_+ \cup S_-$ (the C-legal sample) and k (the bound on the number of misclassifications) be the input for A. Algorithm A proceeds as follows:

1. Let $\epsilon := \frac{1}{|S|+1}$, $\delta := 1/2$, c be any element of C consistent with S, and D be the distribution defined by $D(x) = w(x)/|S|$, where $w(x)$ is the number of occurrences of x in S.

2. Simulate L w.r.t. ϵ, δ, c, D; note that we don't need c itself to do this, since $S_+ = \{x \in c : D(x) \neq 0\}$. Let h be the hypothesis produced by L.

3. Accept iff h makes at most k misclassifications on S.

If each hypothesis of H makes more than k misclassifications on S, then A does not accept (S, k). If there exists a hypothesis $h \in H$ with at most k misclassifications on S, then, with a probability of 1/2 (the confident case), L

outputs a hypothesis with at most $\lfloor k + |S| \cdot \epsilon \rfloor = k$ misclassifications on S. In this case, (S, k) is accepted. □

Corollary 2.3 *If RP $\neq$ NP and MinDis(H) is NP-hard, then H does not allow robust learning.*

Proof. Apply Theorem 2.2 with $C = \mathcal{A}$. □

In the following sections, we investigate concrete hypothesis classes related to separating hyperplanes in the Euclidean space.

3 Robust Learning and Halfspaces

Euclidean concepts (or hypotheses) are subsets of $\mathcal{R}^n$ (the Euclidean n-space augmented with the σ-algebra of Borel sets). *Euclidean examples* are vectors from $\mathcal{R}^n$. Euclidean concepts are quite important in pattern recognition where objects are often described by real feature vectors. From now on, we consider n as a variable complexity parameter of all learning problems associated with Euclidean concepts (although we omit n as an index for the sake of readability).

A well-known method of fixing an Euclidean hypothesis class H is neural network design. The architecture of a net represents H, and the fixing of the net parameters after the learning phase corresponds to the selection of a particular hypothesis $h \in H$. Since simple architectures lead to good generalizations from training examples to test examples, it happens quite often that the representational power of the architecture is overstrained. We call a neural network architecture NNA *robustly trainable* if the associated hypothesis class $H = H(\text{NNA})$ allows robust learning. NNA is called *trainable* if $H(\text{NNA})$ is learnable.

In this section, we discuss the hypothesis class *Halfspaces* of linear halfspaces in $\mathcal{R}^n$ with variable dimension n. Notice that the consistency problem for Halfspaces is just linear programming, which can be done in polynomial time [19, 16]. It is therefore not surprising that Halfspaces is learnable (see [7]). But we state that Halfspaces does not allow robust learning (assumed RP $\neq$ NP). Notice that Halfspaces is the hypothesis class associated with the simplest neural architecture, consisting of a single McCulloch-Pitts neuron only. It follows that a McCulloch-Pitts neuron is trainable, but not robustly trainable (assumed RP $\neq$ NP).

A halfspace is naturally represented by a *separating hyperplane* of the form

$$w \cdot x = w_1 x_1 + \ldots + w_n x_n = \tau.$$

The corresponding *positive* and *negative halfspaces* are given by $w \cdot x \geq \tau$ and $w \cdot x < \tau$, respectively. The unmodified term *halfspace* is used to mean *positive halfspace*. The problem MinDis(Halfspaces) is then the following:

Given disjoint sets of positive and negative examples from $\mathcal{Z}^n$ and a bound $k \geq 1$, does there exist a separating hyperplane which leads to at most k misclassifications?

Note that we restrict the inputs for MinDis(Halfspaces) to be integers (which is basically equivalent to assuming rationals) in order to avoid real arithmetic. It turns out that already this restricted version is NP-hard. We also replaced the notion multiset in the general definition of MinDis(H) above by the notion set, because in the following sections we will show that even this restricted version suffices to prove our results (the only exception is theorem 6.5). It is evident that MinDis(Halfspaces) belongs to NP because there is always a polynomial 'guess' for the (rational) coefficients of an optimal separating hyperplane. We omit the details and refer the reader to standard books on linear programming such as [22].

Theorem 3.1 *MinDis(Halfspaces) is NP-complete.*

We give a proof of this theorem in the next section by proving a strengthened result. A. Blum [5] independently suggested another proof of Theorem 3.1 using a polynomial transformation from the NP-complete problem Open Hemisphere (problem MP6 in [12]) to MinDis(Halfspaces). While his proof needs multisets in the definition of MinDis(Halfspaces), or alternatively uses vectors with non-Boolean entries, our proof will only use sets of Boolean vectors.

Open Hemisphere is just the minimizing disagreement problem for homogeneous halfspaces (halfspaces whose separating hyperplane passes through the origin), so MinDis(Homogeneous Halfspaces) is also NP-complete. Using this fact and Theorem 3.1, we immediately obtain the following corollary:

Corollary 3.2 *If RP $\neq$ NP then*

1. *Halfspaces do not allow robust learning.*
2. *McCulloch-Pitts neurons are not robustly trainable.*
3. *Homogeneous halfspaces do not allow robust learning.*

In Section 4 we will strengthen these results.

4 Limited Degradation and Halfspaces

Under the assumption that RP $\neq$ NP, we have ruled out the possibility that halfspaces allow robust learning (or that McCulloch-Pitts neurons are robustly trainable). Robust learning requires in the confident case that the prediction error of the hypothesis is bounded by $\text{opt}(H) + \epsilon$. We shall consider two ways of relaxing the requirements of robust learning. One is to only require that

we approach some fixed multiple $d \geq 1$ of the minimum error achievable by hypotheses from H. This can still be of practical use if d is close to 1. The other is to enlarge our learning environment. Assume that we have a hypothesis class H', which defines the optimal error $\mathrm{opt}(H')$ achievable using hypotheses in H' to approximate the given concept. To relax the restrictions of the learning problem we may now increase the power of the hypothesis class while preserving the value of $\mathrm{opt}(H')$ as the measure of our learning success. This means we use the class H' as a touchstone class to define the goal of the learning problem, but allow a hypotheses class $H \supseteq H'$, which provides the hypotheses of our learning algorithm. This framework was introduced by Kearns et.al. in [18]. It is especially useful to overcome representational problems inherent to a given touchstone class. We combine these in the following definitions:

- We say that the *H'-degradation* of a learning algorithm L for C, H is *limited by d* if, in the confident case, L outputs a hypothesis $h \in H$ whose prediction error is at most $d \cdot \mathrm{opt}(H') + \epsilon$. Notice that if $C = H' = H$ this still implies the learnability of C (the case $\mathrm{opt}(H') = 0$.)
- We say that the H'-degradation of H is *limited by d when learning concepts from C* if there exists a learning algorithm for C, H whose H'-degradation is limited by d.
- We say that the H'-degradation of H is *unlimited when learning concepts from C* if the degradation is not limited by any fixed d when learning concepts from C.
- We use H and $\mathcal{A}$ as the default values for H' and C respectively, e.g. "the degradation of H is unlimited" means that the H-degradation of H is unlimited when learning concepts from $\mathcal{A}$.

The question of whether the H'-degradation of H is limited is related to the hardness of approximating the minimization problem $\mathrm{MinDis}(H)$. Before stating this connection we present some definitions which will be needed in this section.

- A *minimization problem* Π has the following form, for some pair of predicates P and Q and integer-valued 'cost' function $\mathcal{C}$: Given x satisfying $P(x)$ and an integer bound $k \geq 0$, does there exist a *solution* y satisfying $Q(x, y)$ and $\mathcal{C}(x, y) \leq k$?
- A *polynomial-time approximation algorithm (PTAA)* A for Π is a polynomial-time algorithm which takes as input some x satisfying $P(x)$ and outputs a y satisfying $Q(x, y)$. We write $A(x)$ for the cost of the solution output by A when run on input x.
- The problem of *approximating Π to within a factor d* is the following: Given x satisfying $P(x)$, find some y satisfying $Q(x, y)$ whose cost is at most d times the minimum possible.

- We say that a sample S is (H, H')-*indifferent* if the minimum number of misclassifications on S achievable by hypotheses from H' is the same as the minimum achievable by hypotheses from H.

Theorem 4.1 *Let $H' \subseteq H$. If RP $\neq$ NP and it is NP-hard to approximate MinDis(H), restricted to C-legal and (H, H')-indifferent samples, to within a rational factor d, then the H'-degradation of H is not limited by d when learning concepts from C.*

Proof. Parallels the proof of Theorem 2.2, converting a learning algorithm L for C, H whose H'-degradation is limited by d into a Monte Carlo algorithm for approximating MinDis(H), restricted to C-legal and (H, H')-indifferent samples, to within a factor d. One complication is that we must express d as a ratio of positive integers d_1/d_2, and define ϵ to be $1/(d_2(|S|+1))$. Details are given in [26]. □

The main result of this section is to demonstrate limits on the approximability of MinDis(Halfspaces) and variants. Our tool for doing this is the cost-preserving polynomial transformation:

- A *cost-preserving* polynomial transformation between two minimization problems is one for which instances of the form (I, k) are mapped to instances of the form (I', k), i.e., the cost bound is unchanged. In addition, there must be a polynomial-time algorithm mapping (I, y'), for any solution y' of I', to a solution of I of no greater cost. (For many transformations this mapping is trivial and obvious, and so is not explicitly given.) We write $\Pi \leq_{pol}^{cp} \Pi'$ if there is a cost-preserving polynomial transformation from Π to Π'.

We are now ready to strengthen Theorem 3.1. We will do this by giving a cost-preserving polynomial transformation from Hitting Set to MinDis(Halfspaces).

- **Hitting Set.**
  Input: A finite set T, a collection C of nonempty subsets of T, and integer $k \geq 1$.
  Question: Is there a hitting set for C of cardinality at most k, i.e., a subset R of T s.t. $R \cap M \neq \emptyset$ for all $M \in C$?

Hitting Set is not only NP-complete, it is also difficult to approximate. Bellare, Goldwasser, Lund and Russel [4] have proven the following:

Theorem 4.2 *For all d, no PTAA approximates Hitting Set to within a constant factor d, unless P=NP.*

They actually prove the above for Minimum Set Cover, which is isomorphic to Hitting Set [20, 15]. We now give the transformation from Hitting Set to MinDis(Halfspaces).

Theorem 4.3 *Hitting Set* $\leq_{pol}^{cp}$ *MinDis(Halfspaces).*

Proof. Let (T', C', k) be an instance of Hitting Set. Let s be the cardinality of the largest set in C', or 2 if this is larger. To every $M \in C'$ add $s - |M|$ new, distinct dummy elements, and call the result C. Add to T' all the dummy elements, and call the result T. It is clear that any hitting set for C' is also a hitting set for C. Conversely, given any hitting set R for C we can obtain a hitting set R' for C' s.t. $|R'| \leq |R|$, as follows: replace any dummy element i of R by any non-dummy element of the unique $M \in C$ s.t. $i \in M$. Thus there is a hitting set for C' of size at most k iff there is a hitting set for C of size at most k.

We now construct an instance (S_+, S_-, k) of MinDis(Halfspaces) from (T, C, k), using the fact that all elements of C have the same size $s \geq 2$. Let $T = \{1, \ldots, n\}$. Our example vectors will be of dimension sn, which should be viewed as s groups of n dimensions. We write $\bar{1}_{i_1,\ldots,i_p}$ for the n-dimensional vector having 1 at positions $i_1, \ldots, i_p$ and 0 at all other positions, and $\bar{0}$ for the n-dimensional null vector. We then define

- S_+ is the set of "element vectors" $(\bar{1}_i, \ldots, \bar{1}_i)$, $1 \leq i \leq n$;
- S_- is the set of "set vectors"

  $$(\bar{1}_{i_1,\ldots,i_s}, \bar{0}, \ldots, \bar{0}), \ \ldots \ , (\bar{0}, \ldots, \bar{0}, \bar{1}_{i_1,\ldots,i_s})$$

  for each set $M = \{i_1, \ldots, i_s\} \in C$.

Thus there are n positive vectors and $s|C|$ negative vectors. The theorem follows immediately from

Claim For all r, there exists a hitting set $R \subseteq T$ for C of size at most r iff there exists a separating hyperplane with at most r misclassifications on $S = S_+ \cup S_-$.

The proof of the claim splits into two parts:

$\Longrightarrow$ Given R, we define a hyperplane by $\sum_{j=1}^{s} \sum_{l \in R} -x_{j,l} = 0$. A set vector derived from $M \in C$ is evaluated to $-|R \cap M|$; since $R \cap M \neq \emptyset$, this expression is less than 0. Hence all set vectors are correctly classified as negative. An element vector derived from $i \in T$ is mapped to $-s$ ($i \in R$) or to 0 ($i \notin R$). In the first case the vector is incorrectly classified as negative, and in the second case the vector is correctly classified as positive. This gives $|R| \leq r$ misclassifications altogether.

$\Longleftarrow$ Let $\sum_{j=1}^{s} \sum_{l=1}^{n} w_{j,l} x_{j,l} = \tau$ be a hyperplane which misclassifies at most r vectors of S. We find a corresponding hitting set R as follows:

- If an element vector is misclassified, then insert the corresponding element i into R.

- If a set vector is misclassified, then insert an arbitrary element of the corresponding set M into R.

The resulting set R contains at most r elements. We claim that it is a hitting set. Assume for the sake of contradiction that there exists a set $M \in C$ with $i \notin R$ for all $i \in M$. This implies that the s element vectors corresponding to the elements of M are classified as positive, leading to

$$\forall l \in M, \sum_{j=1}^{s} w_{j,l} \geq \tau, \text{ hence } \sum_{j=1}^{s} \sum_{l \in M} w_{j,l} \geq s\tau,$$

and the s set vectors corresponding to M are classified as negative, leading to

$$\forall 1 \leq j \leq s, \sum_{l \in M} w_{j,l} < \tau, \text{ hence } \sum_{j=1}^{s} \sum_{l \in M} w_{j,l} < s\tau$$

— a contradiction.

□

Note that in the polynomial transformation given above, only Boolean example vectors were produced. We will call the restriction of Halfspaces to Boolean vectors "Boolean Halfspaces." Thus we have

Corollary 4.4 *Hitting Set* $\leq_{pol}^{cp}$ *MinDis(Boolean Halfspaces).*

We now consider the concept class of hyperplanes. Because

$$w \cdot x = \tau \iff w \cdot x \geq \tau \text{ and } -w \cdot x \geq -\tau,$$

these concepts are representable by a very simple net architecture that has one hidden layer containing two neurons. We obtain, however, the following:

Corollary 4.5 *Hitting Set* $\leq_{pol}^{cp}$ *MinDis(Boolean Halfspaces) restricted to Hyperplanes-legal samples.*

Proof. The positive examples in the proof of Theorem 4.3 all lie on the hyperplane $\sum_{l=1}^{n} w_{1,l} = 1$, and none of the negative examples lie on this hyperplane. □

Now let us consider the hypothesis classes of monomials and 1-DL formulas. These are concepts on Boolean vectors. A monomial has the form

$$\{\bar{x} : \gamma_1(x_1) \wedge \cdots \wedge \gamma_n(x_n)\},$$

where each $\gamma_i(x)$ is either "true", "x," or "$\neg x$." A 1-DL formula [24] is defined by a list of pairs $(l_1, c_1) \cdots (l_p, c_p)(l_{p+1}, c_{p+1})$, where each $c_i \in \{0, 1\}$, l_{p+1} is "true", and each l_i $(i \leq p)$ is a literal (either x_j or $\neg x_j$ for some j). We determine if $\bar{x}$ is in the concept by scanning the list from left to right until we find an l_i that is true for $\bar{x}$; then $\bar{x}$ is in the concept iff $c_i = 1$.

Lemma 4.6 *Monomials $\subseteq$ 1-DL $\subseteq$ Boolean Halfspaces.*

Proof. The monomial $\wedge_{i=1}^{p} l_i$ is the same as the 1-DL formula $(\neg l_1, 0) \cdots (\neg l_p, 0)(\text{true}, 1)$. Thus every monomial is also a 1-DL formula.

Now consider the 1-DL formula $L = (l_1, c_1) \cdots (l_p, c_p)(\text{true}, c_{p+1})$. Let $\sigma(1) = 1$ and $\sigma(0) = -1$. Noting that $\neg x_i = (1 - x_i)$, we see that each l_i is a linear function of $\bar{x}$. A straightforward induction on p shows that the halfspace defined by

$$\sum_{i=1}^{p} \sigma(c_i) 2^{p+1-i} l_i \geq -\sigma(c_{p+1})$$

is equivalent to L. Thus every 1-DL formula is also a Boolean halfspace. □

Corollary 4.7 *Hitting Set $\leq_{pol}^{cp}$ MinDis(Boolean Halfspaces) restricted to Hyperplanes-legal, (Boolean Halfspaces, Monomials)-indifferent samples.*

Proof. Returning to the proof of Theorem 4.3, let the instance (T', C', k) of Hitting Set transform to the instance (S_+, S_-, k) of MinDis(Halfspaces). The proof showed for all r that C' has a hitting set of size at most r iff there is a hypothesis $h \in$ Halfspaces with at most r misclassifications on $S = S_+ \cup S_-$. In particular, if r is the size of the smallest hitting set of C', then r is also the minimum number of misclassifications on S achievable by hypotheses from Halfspaces.

It was also shown that if C' has a hitting set R of size r, then the halfspace defined by $\sum_{j=1}^{s} \sum_{l \in R} -x_{j,l} \geq 0$ has at most r misclassifications on S. But this halfspace, when restricted to Boolean vectors, is in fact a monomial, viz., $\bigwedge_{j=1}^{s} \bigwedge_{l \in R} \neg x_{j,l}$. Thus the minimum number of misclassifications on S achievable by monomials is the same as the minimum achievable by halfspaces. □

We have a similar corollary for homogeneous halfspaces:

Corollary 4.8 *Hitting Set $\leq_{pol}^{cp}$ MinDis(Halfspaces) restricted to Hyperplanes-legal, (Halfspaces, Homogeneous Halfspaces)-indifferent samples.*

Proof. Same as the proof of Corollary 4.7, except that this time we note that the halfspace defined by $\sum_{j=1}^{s} \sum_{l \in R} -x_{j,l} \geq 0$ is homogeneous. □

We can readily extend these results to cover the degradation of Monomials, 1-DL formulas, and Homogeneous Halfspaces using the following lemma:

Lemma 4.9 *If $H'' \subseteq H' \subseteq H$, then MinDis($H$) restricted to (H, H'')-indifferent samples $\leq_{pol}^{cp}$ MinDis(H') restricted to (H', H'')-indifferent samples.*

Proof. The required cost-preserving polynomial transformation is just the identity function:

1. If (S, k) is an instance of MinDis(H) restricted to (H, H'')-indifferent samples then the minimum number of misclassifications achievable with H is the same as the minimum achievable with H''; and this the same as the minimum achievable with H' (since $\text{opt}(H) \leq \text{opt}(H') \leq \text{opt}(H'')$).

2. Any solution for MinDis(H') is a hypothesis from $H' \subseteq H$, and hence is also a solution for MinDis(H).

□

Corollary 4.10 *The following problems cannot be approximated to within any constant factor in polynomial time, even when restricted to Hyperplanes-legal samples, unless P=NP:*

1. *MinDis(Boolean Halfspaces), even when restricted to (Boolean Halfspaces, Monomials)-indifferent samples.*

2. *MinDis(1-DL), even when restricted to (1-DL,Monomials)-indifferent samples.*

3. *MinDis(Monomials).*

4. *MinDis(Halfspaces), even when restricted to (Halfspaces, Homogeneous Halfspaces)-indifferent samples.*

5. *MinDis(Homogeneous Halfspaces).*

Proof. (1) follows from Corollary 4.7. (2) and (3) follow from (1), Lemma 4.6, and Lemma 4.9. (4) follows from Corollary 4.8. (5) follows from (4) and Lemma 4.9. □

Corollary 4.11 *Unless $RP = NP$,*

1. *Boolean Halfspaces has unlimited (Monomials-) degradation;*

2. *1-DL has unlimited (Monomials-) degradation;*

3. *Monomials has unlimited degradation;*

4. *Halfspaces has unlimited (Homogeneous Halfspaces-) degradation;*

5. *Homogeneous Halfspaces has unlimited degradation.*

These results hold even when the problem is restricted to learning Hyperplanes.

Proof. Follows directly from Corollary 4.10 and Theorem 4.1. □

The negative results of Corollary 4.11 are rather surprising in light of the fact that there exist polynomial algorithms for learning all of the hypothesis spaces mentioned therein, under the PAC model [7, 25, 24]. By allowing concepts that are not in the hypothesis space, we go from a tractable problem to a problem that is not only intractable, but cannot even be approximated to within any constant factor! This suggests that PAO learning may, in general, be much more difficult than PAC learning.

5 Limited Degradation and Boolean Threshold Functions

We now consider more primitive neurons whose weights w_i (omitting the threshold τ) are all 0 or 1. Such neurons compute Boolean threshold functions; in a sense, they 'learn' which variables are essential, and give the same weight to all essential variables. Pitt and Valiant have shown in [23] that the consistency problem for Boolean threshold functions is NP-complete. They are therefore not learnable (unless RP=NP). Of course, no PTAA A for MinDis(Boolean Threshold Functions) can have a guarantee of the form

$$A(S) \leq f(S) \cdot \mathrm{opt}(S)$$

for some function f (unless P=NP), because it is already NP-hard to ask whether $\mathrm{opt}(S) = 0$. It is therefore more interesting to ask whether there are asymptotic performance guarantees of this form, where we restrict ourselves to input instances S with $\mathrm{opt}(S) \geq c$ and $c > 0$ sufficiently large. It is not hard to show the following result:

Theorem 5.1 *Assume $P \neq NP$, and let $0 \leq \alpha < 1$ and $b, c > 0$. Then no PTAA A for MinDis(Boolean Threshold Functions) can guarantee that*

$$A(S) \leq b|S|^{\alpha}\, opt(S) \quad \textit{whenever} \quad opt(S) \geq c,$$

where $|S|$ denotes the length of the string encoding S under any standard encoding of the relevant data structures.

We omit the proof (which is based on standard padding arguments) and devote the rest of this section to a more structural result. The problem MinDis(Boolean Threshold Functions) belongs to the class MIN PB of minimization problems such that

1. a proposed solution can be checked for feasibility in polynomial time;
2. the cost of any feasible solution can be computed in polynomial time;

3. in polynomial time one can find a feasible solution whose cost is bounded by a polynomial p in the size of the input.

A more formal definition is contained in [15, 20], with the only difference being that they do not require that it be possible to find a feasible solution in polynomial time. We impose this extra requirement in order to restrict ourselves to problems where any computational difficulty is in minimizing, and not in finding a feasible solution.

Theorem 5.2 *MinDis(Boolean Threshold Functions) is MIN PB-complete under cost-preserving polynomial transformations.*

The proof uses a technical variant of the well-known satisfiability problem, called Min SAT and defined as follows:

- **Min SAT.**
  Input: a set U of variables, a set $U' \subseteq U$ of special variables, a collection F of clauses, and an assignment $\varphi_0 : U \longrightarrow \{0,1\}$ satisfying all the clauses in F.
  Objective: find a satisfying assignment $\varphi : U \longrightarrow \{0,1\}$ which minimizes $\sum_{u \in U'} \varphi(u)$ (the number of special variables with assignment 1.)

In the world of MIN PB, Min SAT plays the same role as SAT in the world of decision problems: It is a sort of hardest minimization problem. The following result is therefore not surprising.

Lemma 5.3 *Min SAT is MIN PB-complete under cost-preserving polynomial transformations.*

Proof. The proof is an adaption of analogous reasonings in [21, 9], which in turn are adaptions of Cook's theorem [8]. Given an instance x of a MIN PB problem, we first compute a feasible solution y_0 for x. Then we construct a Turing machine M that halts only if its input y is a feasible solution for x of cost at most $p(|x|)$, in which case it computes the cost of its input and writes this cost in unary on a reserved portion of the tape. Cook's construction is then applied to transform M into a set of variables and clauses, and M is simulated on input y_0 to obtain a satisfying assignment. □

The MIN PB-completeness of MinDis(Boolean Threshold Functions) follows now directly from

Lemma 5.4 *Min SAT* $\leq^{cp}_{pol}$ *MinDis(Boolean Threshold Functions).*

Proof. Given m clauses $C_1, \ldots, C_m$ over n variables $x_1, \ldots, x_n$, the set $x_{i_1}, \ldots, x_{i_s}$ of special variables, and a satisfying assignment φ_0, we will construct a corresponding Boolean input sample S for Boolean threshold functions.

For all $1 \leq j \leq m$, let $\bar{a}_j \in \{0,1\}^n$ denote the characteristic vector of C_j, i.e., for all $1 \leq i \leq n$ the i-th component of $\bar{a}_j$ is 1 iff non-negated variable x_i occurs in C_j. Let $\bar{b}_j \in \{0,1\}^n$ denote the corresponding vector w.r.t. negated variables x'_i. Each vector from S will have the form $(\bar{a}, \bar{b}, \bar{c}, \bar{d})$, where $\bar{a}, \bar{b}, \bar{d} \in \{0,1\}^n$ and $\bar{c} \in \{0,1\}^{2n}$. The first $2n$ dimensions correspond to $x_1, \ldots, x_n, x'_1, \ldots, x'_n$. The last $2n+n$ dimensions serve for a gadget construction.

The task of the gadget is mainly to force the threshold of each 'reasonable' threshold function f to be 2, and its binary weights in the c part to be 1. Function f is said to be *reasonable* on the sample S if it produces fewer than n misclassifications. Notice that for each unreasonable functions f, there is always the trivial assignment φ_0 which has cost at most n (since the number of special variables is bounded by n.) The negative vectors of the gadget are

$$(\bar{0}, \bar{0}, \bar{1}_k, \bar{0}),$$

and the positive ones

$$(\bar{0}, \bar{0}, \bar{1}_{k',k''}, \bar{0}),$$

for all $1 \leq k \leq 2n$ and $1 \leq k' < k'' \leq 2n$. Threshold 0 is impossible because all $2n$ negative examples would be classified positive. Symmetrically, thresholds exceeding 2 are impossible. Threshold 1 leads either to at least n positively classified negative examples (if at least n weights in the c part are 1) or to at least $\binom{n}{2} \geq n$ (n sufficiently large) negatively-classified positive examples (if at least n weights in the c part are 0). The threshold of each reasonable threshold function is therefore definitely 2. This in turn implies that all weights in the c part are 1 because, otherwise, at least $2n-1 \geq n$ positive examples are classified as negative. This completes the construction of the gadget (the significance of the d part will be revealed in a moment.)

A consistent assignment of the $2n$ literals can now be forced by the positive examples

$$(\bar{1}_i, \bar{1}_i, \bar{1}_k, \bar{0}) \text{ where } (1 \leq i, k \leq n),$$

ruling out $x_i = x'_i = 0$, and the negative examples

$$(\bar{1}_i, \bar{1}_i, \bar{0}, \bar{1}_l), \text{ where } (1 \leq i, l \leq n),$$

ruling out $x_i = x'_i = 1$. Note that the running indices l and k create a virtual multiplicity of n for the positive and negative examples, respectively, and therefore force reasonable functions to classify all these examples correctly.

Similarly, a satisfying assignment can be forced by the following positive examples:

$$(\bar{a}_j, \bar{b}_j, \bar{1}_k, \bar{0}) \text{ where } 1 \leq j \leq m, 1 \leq k \leq n.$$

The final component of our construction is a counter for special variables set to 1. It is implemented by the following negative examples:

$$(\bar{1}_{i_\sigma}, \bar{0}, \bar{1}_1, \bar{0}) \text{ where } 1 \leq \sigma \leq s.$$

Note that the c part is here not used for creating a virtual multiplicity, because we want the numbers of special variables with assignment 1 and of misclassified examples to be equal.

It should be evident from the construction that the clauses have a satisfying assignment with at most r special variables set to 1 iff there is a Boolean threshold function for sample S with at most r misclassifications. This completes the cost-preserving transformation. □

6 Heuristical Learning and Single Neurons

Let $H \subseteq \mathcal{A}$ be a hypothesis class which does not allow robust learning. Then concept classes like $C = \mathcal{A}$ (and probably also some smaller classes) cannot be PAO learned by H (unless RP = NP). It is, however, still possible that good 'rules of thumb' can efficiently be selected from H for given concepts. More precisely, we might find an $h \in H$ containing almost no negative examples, but covering almost a constant fraction of the positive ones. This leads to decisions with very trustworthy 'yes' answers, but overcareful 'no' answers.

A more formal approach to these ideas is based on Valiant's notion of heuristical learning (see [23]). As with PAC learning, we have some unknown concept c; we also have unknown distributions D_+ and D_- on c (the positive examples) and $\bar{c}$ (the negative examples), respectively, because it will be technically convenient to have two separate distributions for the two types of examples.

- A *heuristical learning algorithm L with accept rate r* ($0 < r < 1$, $r \in \mathcal{Q}$) for concept class C and hypothesis class H is defined as follows:
  Input: Two parameters $0 < \epsilon, \delta < 1$ and a sample of $m_+ = m_+(\epsilon, \delta)$ positive and $m_- = m_-(\epsilon, \delta)$ negative examples (drawn according to $D_+^{m_+}$ and $D_-^{m_-}$, respectively). We demand that m_+ and m_- be polynomially bounded in $1/\epsilon, 1/\delta$.
  Objective: After polynomially many steps, L must ouput a rule $h \in H$ which satisfies with a probability of at least $1 - \delta$ (the confident case) the following condition:

  $$D_-(h \cap \bar{c}) \leq \epsilon \text{ and } D_+(h \cap c) \geq r - \epsilon.$$

  If however no rule from H satisfies

  $$D_-(h \cap \bar{c}) = 0 \text{ and } D_+(h \cap c) \geq r,$$

  then L may output an error message.

The requirements for L must be satisfied for all $c, \epsilon, \delta, D_+, D_-$. Notice that ϵ is part of the input, whereas r is a fixed constant. In the confident case, the probability that a negative example for c is correctly classified can be made

arbitrarily close to 1, whereas positive examples may always be misclassified with a probability of $1 - r + \epsilon$.

- We say that C is *heuristically learnable by H with accept rate r* if a heuristical learning algorithm with accept rate r for C and H exists.

These definitions generalize to the parameterized case in the obvious way.

There is again a combinatorial problem, called $Rule(C, H, r)$, which is strongly related to heuristical learnability:

- **Rule**(C, H, r).
  Input: A C-legal sample $S = S_+ \cup S_-$ (disjoint union of multisets.)
  Question: Is there a rule $h \in H$ that is consistent with S_- (i.e., $S_- \subseteq \bar{h}$) and covers at least a fraction r of S_+ (i.e., $|S_+ \cap h| \geq r \cdot |S_+|$)?

The following result is from [23]:

Theorem 6.1 *If $RP \neq NP$ and $Rule(C, H, r)$ is NP-hard, then C is not heuristically learnable by H with accept rate r.*

One main result of this section is the following.

Theorem 6.2 *For all rationals $0 < r < 1$, Rule(Hyperplanes,Halfspaces,r) is NP-complete.*

Proof. The proof uses restricted versions of Vertex Cover called s-Ratio Vertex Cover for all fixed rational constants $0 < s < 1$:

- **Vertex Cover.**
  Input: A graph $G = (V, E)$ and an integer bound $k \geq 0$.
  Question: Does there exist a set $U \subseteq V$ (called the *vertex cover*) such that $|U| \leq k$ and for each edge $\{u, v\} \in E$ at least one of u and v belongs to U?

- **s-Ratio Vertex Cover.**
  Input: A graph $G = (V, E)$ with $n = |V|$ vertices.
  Question: Does there exist a vertex cover of G of size at most sn?

Let $G = ((V, E), k)$ be an arbitrary input instance of Vertex Cover. WLOG assume that $k \leq sn + 1$ (this constraint can always be satisfied by adding artificial isolated vertices to G). We now obtain a polynomial transformation to s-Ratio Vertex Cover by adding a disjoint N-clique to G for a properly chosen N. The new number of vertices is $n + N$. The new bound on the size of the vertex cover is $k + N - 1$ because the minimum vertex cover of an N-clique has size $N - 1$. Ideally, N should satisfy $k + N - 1 = s(n + N)$. N can be chosen as the integer part of the solution of this equality, i.e., $N = \lfloor (sn - k + 1)/(1 - s) \rfloor$. This transformation works properly, if $N \geq 0$, which holds because $k \leq sn + 1$.

Let $0 < r < 1$ be a fixed rational constant and $s = 1 - r$. We now construct an instance (S_+, S_-, k) of Rule(Hyperplanes,Halfspace,r) from an instance $G = (V, E)$ of s-Ratio Vertex Cover, as follows:

- S_+ is the set of "vertex examples" $\bar{1}_i$, $1 \leq i \leq n = |V|$.
- S_- is the vector $\bar{0}$ together with the set of "edge examples" $\bar{1}_{i,j}$, for all $\{v_i, v_j\} \in E$.

Note that all positive examples lie on the hyperplane $\sum_i x_i = 1$, and none of the negative examples lie on this hyperplane; thus this is a Hyperplanes-consistent sample.

If $U \subseteq V$ is a vertex cover of size $k \leq sn$, then consider the halfspace $w \cdot x \geq 1$, where $w_i = -1$ if $v_i \in U$ and $w_i = 1$ otherwise. It is easy to verify that this halfspace misclassifies exactly the k examples corresponding to vertices from U.

For the reverse direction, assume that $w \cdot x \geq \tau$ is a halfspace that misclassifies $k \leq (1-r)n$ vertex examples and is consistent with the negative examples. We insert a vertex v_i into the vertex cover U iff $\bar{1}_i$ is misclassified. Thus, $|U| = k$. Note that for each edge $\{v_i, v_j\}$, at least one of $\bar{1}_i, \bar{1}_j$ is misclassified, since $\bar{0}$ and $\bar{1}_{i,j}$ are both classified as negative. Thus U is a vertex cover for G. It follows that there exists a vertex cover with at most $sn = (1-r)n$ vertices iff there exists a halfspace covering at least $(1-s)n = rn$ positive (and no negative) examples. This shows that Rule(Hyperplanes,Halfspaces,r) is NP-complete. □

Corollary 6.3 *If RP $\neq$ NP, then hyperplanes (or the hyperplane architecture) are not heuristically learnable by halfspaces (or McCulloch-Pitts neurons) with a fixed constant accept rate.*

In the second part of this section, we consider *heuristical learning with constant reject rates* r $(0 < r < 1, r \in \mathcal{Q})$. A *hypothesis (or rule) with reject* is a partition of X into 3 disjoint sets h_+, h_-, h_{rej}. In the following condition it is again more convenient to deal with a common distribution D for both types of examples. We demand that the outputs of the corresponding learning algorithms satisfy

$$D(h_+ \cap \bar{c}) + D(h_- \cap c) \leq \epsilon \text{ and } D(h_{\text{rej}}) \leq r + \epsilon.$$

If however no rule from H satisfies

$$D(h_+ \cap \bar{c}) + D(h_- \cap c) = 0 \text{ and } D(h_{\text{rej}}) \leq r,$$

then the output may be an error message. It should be clear without a detailed definition what we mean by heuristical learning algorithms with reject rate r for concept class C and hypothesis class H (the omitted details are quite similar to the definition of heuristical learning algorithms with accept rate r.)

- We say that C is *heuristically learnable by H with reject rate r* if an algorithm of the kind described above exists.

Again, the generalization to the parameterized case is obvious. We also define a related combinatorial problem:

- **Reject-Rule**(C, H, r).
  Input: A C-legal sample $S = S_+ \cup S_-$ (disjoint union of multisets.)
  Question: Is there a rule $h = (h_+, h_-, h_{\text{rej}})$ such that

$$h_+ \cap S_- = \emptyset, h_- \cap S_+ = \emptyset \text{ and } |h_{\text{rej}} \cap S| \leq r \cdot |S|?$$

Reasonings analogous to those of Section 2 lead to the following.

Theorem 6.4 *If $RP \neq NP$ and Reject-Rule(C, H, r) is NP-hard, then C is not heuristically learnable by H with reject rate r.*

We now define a new class of hypotheses:

- A *(closed) stripe* is the intersection of two (closed) halfspaces whose defining hyperplanes are parallel. Each stripe Σ partitions the Euclidean space into Σ and two open halfspaces Σ_+, Σ_-.

- *Reject-Stripes* is the class of Euclidean rules (with reject) of the form $(\Sigma_+, \Sigma_-, \Sigma)$ where the reject region Σ is a stripe (see Figure 1.)

Reject stripes correspond to *sigmoidal neurons*. These neurons compute $f(w \cdot x)$ for some continuous function f which strictly monotonously increases from 0 to 1. They are naturally interpreted as rules with reject if we identify outputs below a certain threshold α with 0, outputs above a certain threshold $\beta > \alpha$ with 1, and the remaining outputs with 'reject'. The resulting class of reject regions is just the class of stripes.
The second main result in this section is the following.

Theorem 6.5 *For all rational $0 < r < 1$:*
Reject-Rule(Hyperplanes,Reject-Stripes,r) is NP-complete.

Proof. We give a polynomial transformation from Vertex Cover to Reject-Rule(Hyperplanes, Reject-Stripes, r). Let $(G = (V, E), k)$ be an instance of Vertex Cover, and let $n = |V|$. As in the proof of Theorem 6.2, we may assume WLOG that $k \leq rn$. Let $G' = (V', E')$ be the graph obtained from G by adding a disjoint N-clique, where N will be defined later, and let V_c be the vertices of this N-clique. We now construct an instance of Reject-Rule(Hyperplanes, Reject-Stripes, r), using essentially the same examples as in the proof of Theorem 6.2:

- S_+ is the multiset containing t copies of each vector $\bar{1}_i$, $1 \leq i \leq n + N$ (t will be defined later.)

- S_- is the vector $\bar{0}$ together with the set of $\bar{1}_{i,j}$, for all $\{v_i, v_j\} \in E'$.

As before, this is a Hyperplanes-consistent sample. In what follows, let $m = |E| + 1$ and $q = t(n+N) + m + \binom{N}{2}$ (the total number of examples.)

Suppose that U is a vertex cover of G, of size at most k. Then for all $v \in V_c$, $U' = U \cup V_c \setminus \{v\}$ is a vertex cover of G', of size at most $k+N-1$. Defining $w_i = 1$ if $v_i \notin U'$ and $w_i = -1$ if $v_i \in U'$, the reject stripe Σ given by $-2 \leq w \cdot x \leq 0$ correctly classifies all positive examples $\bar{1}_i$, $v_i \notin U'$, and rejects the remaining $B_1 = t(k+N-1) + m + \binom{N}{2}$ examples. We thus need $B_1 \leq rq$.

For the reverse direction, suppose that Σ is a reject stripe containing at most rq examples and being consistent with all remaining examples. We define U' to be set of v_i for which $\bar{1}_i \in \Sigma$. Then $|U'| \leq rq/t$. We claim that for each edge $\{v_i, v_j\} \in E'$, at least one of $\bar{1}_i, \bar{1}_j$ must belong to Σ. Assume for the sake of contradiction that $\bar{1}_i, \bar{1}_j \notin \Sigma$. Let Σ_+, Σ_- be the two halfspaces outside Σ. Then $\bar{1}_i, \bar{1}_j \in \Sigma_+$ and $\bar{0}, \bar{1}_{i,j} \in \Sigma \cup \Sigma_-$ — a contradiction to the convexity of Σ_+ and $\Sigma \cup \Sigma_-$. So U' is a vertex cover of G' of size at most $\lfloor rq/t \rfloor$. Since any vertex cover of G' contains at least $N-1$ elements of V_c, U' contains a vertex cover of G of size at most $\lfloor rq/t \rfloor - N + 1$. We thus need $k+1 > rq/t - N + 1$, i.e. $rq < t(k+N) = B_2$.

The theorem will be proven if we can find choices for t and N that satisfy $B_1 \leq rq < B_2$. Applying some algebra, these two inequalities are rewritten as

$$\frac{1}{1-r}\left(rn - k + r\left(m + \binom{N}{2}\right)/t\right) < N \leq \frac{1}{1-r}(rn-k+1) - \left(m + \binom{N}{2}\right)/t.$$

Setting

$$t = \left\lceil \frac{2}{r} \cdot \max\left\{m, \binom{N}{2}\right\} \right\rceil$$

it becomes sufficient to satisfy

$$B_3 := \frac{1}{1-r}(rn-k+rm/t) + \frac{r^2}{2(1-r)} < N \leq \frac{1}{1-r}(rn-k+1) - \frac{m}{t} - \frac{r}{2} =: B_4.$$

An easy calculation shows that $B_4 - B_3 \geq 1$. Setting $N = \lfloor B_4 \rfloor$, we obtain a successful polynomial transformation from Vertex Cover to Reject-Rule(Hyperplanes,Reject-Stripes,r). □

Corollary 6.6 *If RP ≠ NP, then hyperplanes are not heuristically learnable by reject stripes (or the sigmoidal neuron) with a fixed, constant reject rate.*

7 Conclusions and Open Problems

The results of this paper show that the learning performance of single neurons degrades rapidly when their representational power is overstrained (assuming

RP $\neq$ NP). It appears likely that the same phenomenon occurs in larger networks. There are two possible conclusions.

One conclusion is that a learning algorithm should not run on a fixed architecture. It should expand the architecture when overstraining occurs, or contract it if there is evidence for understraining. This coincides with the empirical observation of many researchers and their heuristical attempts to specify appropriate rules for expansion or contraction (see [10, 11]). We may consider our results as complexity-theoretical support for their approaches. It is also interesting to note that the concept class we have used for knocking out a single neuron (hyperplanes) is PAC learnable. Two hidden units in one hidden layer are sufficient to represent hyperplanes. Thus, learning tasks which are hard for a single neuron can be quite easy for slightly stronger architectures.

Another conclusion is that our learning model might be too pessimistic or too restrictive: too pessimistic in the sense that worst-case concepts or worst-case distributions do (hopefully) not occur in practice; too restrictive in the sense that learning should use more powerful tools than random examples only. It is however not quite clear which additional tools are available for the purposes of neural learning.

Our notions of learnability are not new, except for limited degradation and heuristical learning with constant reject rate. Robust learning has already been used by Abe, Takeuchi and Warmuth in [1, 2]. Their notion of robust learning includes also stochastic rules (without a deterministic distinction between positive and negative examples) and allows cost measures different from the prediction error. Notice however that negative results are stronger for the more specific situation, whereas positive results are stronger for the general case. Therefore, we did not introduce the notion of robust learning in full generality. Heuristical learning with constant accept rate has been introduced by Pitt and Valiant in [23]. They also applied the combinatorial method to PAC learning and heuristical learning. In [2], this method is used for showing that probabilistic automata are not robustly trainable.

There exist some other nonlearnability results concerning neural nets, which we will mention briefly. Judd proved various nonlearnability results in this context (see [14]). His polynomial transformations from NP-hard problems to consistency problems suffer a little bit from using quite artificial architectures which are not likely to be used in any existing learning environment. Nevertheless, his results are among the first rigorously proven negative results concerning neural learning. Blum and Rivest in [6] showed nonlearnabilty for quite simple architectures: one hidden layer with $k \geq 2$ hidden units and one output unit which computes the logical AND. This architecture represents polyhedrons with k facets, i.e., intersections of k halfspaces. Our paper followed this tendency of simplification by investigating single neurons: the simplest architecture (we guess).

Finally, we would like to mention that our negative results are turned to the positive side when the dimension n of the Euclidean space is fixed. The approxi-

mation problems are then solvable in polynomial time if we apply the algorithm of Johnson and Preparata for the Densest Hemisphere problem (see [13]).

As for future work, we suspect that the phenomenon of unlimited degradation occurs also for heuristical learning. The analysis of learning algorithms on dynamical architectures seems to be one of the most striking problems for future research on neural learning.

References

[1] N. Abe, J. Takeuchi, and M. K. Warmuth, Polynomial learnability of probabilistic concepts with respect to the Kullback-Leibler divergence, *in* "Proceedings of the 4th Annual Workshop on Computational Learning Theory, 1991," pp. 277–289.

[2] N. Abe and M. K. Warmuth, On the computational complexity of approximating distributions by probabilistic automata, *in* "Proceedings of the 3rd Annual Workshop on Computational Learning Theory, 1990," pp. 52–66.

[3] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy, Proof verification and intractability of approximation problems, *in* "Proceedings of the 33rd Annual IEEE Symposium on Foundations of Computer Science, 1992," pp. 14–23.

[4] M. Bellare, S. Goldwasser, C. Lund, & A. Russell, Efficient probabilistically checkable proofs and applications to approximation, *in* "Proceedings of the 25th Annual ACM Symposium on Theory of Computing," pp. 294–304.

[5] A. Blum, personal communication, 1992.

[6] A. Blum and R. L. Rivest, Training a 3-node neural network is NP-complete, *in* "Proceedings of the 1988 Workshop on Computational Learning Theory," pp. 9–18.

[7] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth, Learnability and the Vapnik-Chervonenkis dimension, *J. Assoc. Comput. Mach.* 36 (1989), 929–965.

[8] S. A. Cook, The complexity of theorem-proving procedures, *in* "Proceedings, 3rd Annual ACM Symposium on Theory of Computing, 1971," pp. 151–158.

[9] P. Crescenzi and A. Panconesi, Completeness in approximation classes, *Inform. and Comput.* 93 (1991), 241–262.

[10] Y. L. Cun, J. S. Denker, and S. A. Solla, Optimal brain damage, *in* "Advances in Neural Information Processing Systems 2" (D. S. Touretzky, Ed.), pp. 598–605, Morgan Kaufmann, San Mateo, CA, 1990.

[11] S. E. Fahlman AND C. Lebiere, The cascade-correlation learning architecture, *in* "Advances in Neural Information Processing Systems 2" (D. S. Touretzky, Ed.), pp. 524–532, Morgan Kaufmann, San Mateo, CA, 1990.

[12] M. R. Garey and D. S. Johnson, "Computers and Intractability: A Guide to the Theory of NP-Completeness," Freeman, New York, 1979.

[13] D. S. Johnson and F. P. Preparata, The densest hemisphere problem, *Theor. Comp. Sci.* 6 (1978), 93–107.

[14] J. S. Judd, "Neural Network Design and the Classification of Learning," MIT Press, Cambridge, MA, 1990.

[15] V. Kann, "On the Approximability of NP-complete Optimization Problems," Ph.D. thesis, Royal Institute of Technology, Dept. of Numerical Analysis and Computing Science, Stockholm, Sweden, 1992.

[16] N. Karmarkar, A new polynomial time algorithm for linear programming, *Combinatorica* 4 (1984), 373–395.

[17] M. J. Kearns, M. Li, L. Pitt, and L. Valiant, On the learnability of boolean formula, *in* "Proceedings, 19th ACM Symposium on Theory of Computing, 1987," pp. 285–295.

[18] M. J. Kearns, R. E. Schapire, and L. M. Sallie, Toward efficient agnostic learning, *in* "Proceedings of the 5th Annual Workshop on Computational Learning Theory, 1992," pp. 341–353.

[19] L. G. Khachian, A polynomial algorithm for linear programming, *Soviet Math. Doklady* 20 (1979), 191–194.

[20] P. G. Kolaitis and M. N. Thakur, Approximation properties of NP minimization classes, *in* "Proceedings of the 6th Annual Conference on Structures in Complexity Theory, 1991," pp. 353–366.

[21] P. Orponen and H. Mannila, On approximation preserving reductions: Complete problems and robust measures, Research Report C–1987–28, University of Helsinki, 1987.

[22] C. H. Papadimitriou and K. Steiglitz, "Combinatorial Optimization," Prentice-Hall, Englewood Cliffs, NJ, 1982.

[23] L. Pitt and L. G. Valiant, Computational limitations on learning from examples, *J. Assoc. Comput. Mach.* 35 (1988), 965–984.

[24] R. L. Rivest, Learning decision lists, *Machine Learning* 2 (1987), 229–246.

[25] L. G. Valiant, A theory of the learnable, *Communications of the ACM* 27 (1984), 1134–1142.

[26] K. S. Van Horn, "Learning as Optimization," Ph.D. dissertation, Computer Science Department, Brigham Young University, 1994.