In F. Fogelman-Soulié and P. Gallinari, editors, *Proceedings of the International Conference on Artificial Neural Networks ICANN'95*, volume 1, pages 211–216, EC2 & Cie., Paris, 1995.

On Methods to Keep Learning Away from Intractability

(Extended Abstract)

Michael Schmitt

Institute for Theoretical Computer Science, Technische Universität Graz
Klosterwiesgasse 32/2, A–8010 Graz, Austria, mschmitt@igi.tu–graz.ac.at

Abstract

We investigate the complexity of learning from restricted sets of training examples. With the intention to make learning easier we introduce two types of restrictions that describe the permitted training examples. The strength of the restrictions can be tuned by choosing specific parameters. We ask how strictly their values must be limited to turn NP-complete learning problems into polynomial-time solvable ones. Results are presented for Perceptrons with binary and arbitrary weights. We show that there exist bounds for the parameters that sharply separate efficiently solvable from intractable learning problems.

1 Introduction

Since the advent of learning algorithms for neural networks many problems have been shown computationally intractable. References [1, 4, 10, 11, 14, 19] are just to mention a few. In order to cope with complexity various methods have been invented to incorporate knowledge about the functions being learned into the algorithm. Techniques to modify architectures (see, e.g., [8, Section 6.17] and [9, Section 6.6] for comprehensive descriptions) or to initialize weights [13] are designed to reduce learning times. Their success, however, can only be stated *after* completion of the training process.

In this paper we pursue a different approach. We try to bypass intractability of training a specific architecture by restricting the set of permitted training examples. To this end, we propose two parameters to characterize what makes example sets hard to learn. The first parameter is called "heaviness", the second "coincidence". By heaviness we measure the maximum number of non-zero components of any training example; coincidence

is the maximum number of non-zero components that any two training examples have in common. Bounding heaviness and coincidence of example sets by constants we obtain classes of learning problems whose complexity might depend on the values of these constants: If heaviness is small then a possible adversary can choose examples only from a very restricted set, if coincidence is small then the examples are unlikely to contradict each other. Because heaviness and coincidence of an example set can be computed in linear respectively quadratic time they may provide simple means to separate feasible from intractable instances of learning problems.

The results presented in this paper concern Perceptrons (threshold gates, McCulloch-Pitts neurons) computing Boolean functions with binary weights and arbitrary weights respectively. We show that the problem of achieving agreement with all examples (consistency for short) remains NP-complete for binary weights if the examples are allowed to have heaviness at least 4 and coincidence at least 1. The complementary cases, however, can be solved in linear time. For the problem of minimizing disagreement (maximum consistency for short) we find NP-completeness in case of binary weights already for heaviness 2 and coincidence 1 whereas the complementary problems can be solved in linear time. The latter result also holds if the weights are allowed to be arbitrary real numbers. Finally, total agreement for arbitrary weights is widely known to be computable in polynomial time by methods of linear programming [12]. Thus we are able to completely characterize all defined restrictions of learning problems for this architecture with respect to their complexity.

In Section 2 we give definitions of example properties and learning problems and present the main results. In Section 3 we sketch the NP-completeness results. Linear-time algorithms and sketches thereof are presented in Section 4. The last section contains some concluding remarks and discussions. Proofs in this paper are sketched very coarsely or omitted at all. The interested reader is referred to a more comprehensive technical report [18].

2 Learning from restricted example sets

2.1 Definitions

Given a set of examples $S = \{(x^{(1)}, a^{(1)}), \ldots, (x^{(m)}, a^{(m)})\}$ where $x^{(i)} \in \{0,1\}^n$ and $a^{(i)} \in \{0,1\}$, we denote by $\mathrm{pos}(S) = \{x \mid (x,1) \in S\}$ the set of *positive* examples and by $\mathrm{neg}(S) = \{x \mid (x,0) \in S\}$ the set of *negative* examples. For the sake of generality we allow example sets to contain contradictions. The set S is said to be *consistent* if $\mathrm{pos}(S) \cap \mathrm{neg}(S) = \emptyset$. The *domain* of S is defined as $\mathrm{dom}(S) = \mathrm{pos}(S) \cup \mathrm{neg}(S)$. Sets of examples are considered as instances of decision problems. To restrict the set of possible instances we introduce two properties. *Coincidence* $\zeta(S)$ and *heaviness* $\eta(S)$ of an example set S

are defined as

$$\begin{aligned}\zeta(S) &= \max\{x \cdot y \mid (x,a),(y,b) \in S,\ (x,a) \neq (y,b)\},\\ \eta(S) &= \max\{x \cdot x \mid (x,a) \in S\},\end{aligned}$$

where $x \cdot y$ denotes the inner product. In case $|S| \leq 1$ we define $\zeta(S) = 0$, in case $S = \emptyset$ we let $\eta(S) = 0$. Thus, $\zeta(S)$ is the maximum number of components in which any two elements of the domain have a one in common, and $\eta(S)$ is the maximum Hamming-weight, or number of ones, of the elements. Note that bounded heaviness is a stronger restriction than bounded coincidence because of $\zeta(S) \leq \eta(S)$.

A function f is said to be *consistent with* an example set S if $f(x) = a$ holds for all $(x,a) \in S$. The *consistency problem for a set of functions* $\mathcal{F}$ is the problem to decide if for a given set of examples S there exists a function $f \in \mathcal{F}$ that is consistent with S. Alternative terms frequently used in the literature are training problem [4], loading problem [11], and fitting problem [15]. The name consistency problem often occurs in the context of Valiant's PAC-learning paradigm [2, 5]. Let $\mathcal{B}$ be the set of Boolean functions computable by single neurons (Perceptrons, threshold gates) with binary weights and arbitrary threshold. Further, let C and H be arbitrary natural numbers. The classes of consistency problems for $\mathcal{B}$ with bounded coincidence and heaviness are defined as follows.

CONSISTENCY FOR $\mathcal{B}$ WITH COINCIDENCE C
INSTANCE: Example set S with $\zeta(S) \leq C$.
QUESTION: Is there a function $f \in \mathcal{B}$ that is consistent with S?

CONSISTENCY FOR $\mathcal{B}$ WITH HEAVINESS H
INSTANCE: Example set S with $\eta(S) \leq H$.
QUESTION: Is there a function $f \in \mathcal{B}$ that is consistent with S?

We also consider the combined problem CONSISTENCY FOR $\mathcal{B}$ WITH COINCIDENCE C AND HEAVINESS H which has only sets with $\zeta(S) \leq C$ and $\eta(S) \leq H$ as instances.

The *maximum consistency problem for a set of functions* $\mathcal{F}$ (also known as minimizing disagreement problem) has as instance a set of examples S and a natural number K. The question is whether there is a subset of S containing at least K elements and a function $f \in \mathcal{F}$ consistent with that subset. For natural numbers C and H we define maximum consistency problems for $\mathcal{B}$ as follows.

MAXIMUM CONSISTENCY FOR $\mathcal{B}$ WITH COINCIDENCE C
INSTANCE: Example set S with $\zeta(S) \leq C$, natural number K.
QUESTION: Is there a subset $S' \subseteq S$, $|S'| \geq K$
and a function $f \in \mathcal{B}$ that is consistent with S'?

MAXIMUM CONSISTENCY FOR $\mathcal{B}$ WITH HEAVINESS H
INSTANCE: Example set S with $\eta(S) \leq H$, natural number K.
QUESTION: Is there a subset $S' \subseteq S$, $|S'| \geq K$
and a function $f \in \mathcal{B}$ that is consistent with S'?

Again, we combine both to MAXIMUM CONSISTENCY FOR $\mathcal{B}$ WITH COINCIDENCE C AND HEAVINESS H. We also investigate consistency problems for $\mathcal{L}$, the set of Boolean functions computable by single neurons (Perceptrons, threshold gates) with arbitrary real weights. They are defined correspondingly by replacing $\mathcal{B}$ by $\mathcal{L}$ in the above definitions.

2.2 Main results

Before we state our results for the restricted learning problems we recall known results for the unrestricted versions from the literature. Consistency for $\mathcal{B}$ has been shown NP-complete by Pitt and Valiant [16, Section 5]. This implies that maximum consistency for $\mathcal{B}$ is also NP-complete. Consistency for $\mathcal{L}$ is well known to be solvable in polynomial time by methods of linear programming [12]. Amaldi [1] has shown that maximum consistency for $\mathcal{L}$ with threshold $t = 0$ is NP-complete. The constraint $t = 0$ was eliminated by Höffgen and Simon [10]. We now state our results in the form of so-called dichotomy theorems.

Theorem 1 CONSISTENCY FOR $\mathcal{B}$ WITH COINCIDENCE C AND HEAVINESS H *is* NP*-complete whenever* $C \geq 1$ *and* $H \geq 4$. *On the other hand,* CONSISTENCY FOR $\mathcal{B}$ WITH COINCIDENCE 0 *is solvable in linear time, as well as* CONSISTENCY FOR $\mathcal{B}$ WITH HEAVINESS H *for all* $H \leq 3$.

Theorem 2 MAXIMUM CONSISTENCY FOR $\mathcal{B}$ WITH COINCIDENCE C AND HEAVINESS H *is* NP*-complete whenever* $C \geq 1$ *and* $H \geq 2$. *On the other hand,* MAXIMUM CONSISTENCY FOR $\mathcal{B}$ WITH COINCIDENCE 0 *is solvable in linear time, as well as* MAXIMUM CONSISTENCY FOR $\mathcal{B}$ WITH HEAVINESS H *for all* $H \leq 1$.

Theorem 3 MAXIMUM CONSISTENCY FOR $\mathcal{L}$ WITH COINCIDENCE C AND HEAVINESS H *is* NP*-complete whenever* $C \geq 1$ *and* $H \geq 2$. *On the other hand,* MAXIMUM CONSISTENCY FOR $\mathcal{L}$ WITH COINCIDENCE 0 *is solvable in linear time, as well as* MAXIMUM CONSISTENCY FOR $\mathcal{L}$ WITH HEAVINESS H *for all* $H \leq 1$.

The reader will notice that Theorem 3 is a replica of Theorem 2 word for word except for the letter $\mathcal{B}$ which is replaced by $\mathcal{L}$. Interestingly, Theorem 1 does not have such a replica due to the fact that linear programming can be done in polynomial time. All defined restrictions of learning problems are now completely characterized with respect

to their complexity. One should have in mind that the dichotomies are proper only if P $\neq$ NP. Some proof sketches for the above theorems are given in the following two sections. Section 3 is devoted to the lower bounds, Section 4 presents algorithms and upper bounds. Theorem 1 will then follow from Theorem 5, Lemma 8 and Theorem 9, Theorem 2 will be shown by Theorem 6 and Theorem 10, and Theorem 3 will be a consequence of Corollary 7 and Corollary 11.

3 NP-Complete learning problems

For the NP-completeness result stated in Theorem 1 we use a stronger version of a particular satisfiability problem which we call ALMOST DISJOINT POSITIVE 1-IN-3SAT.

ALMOST DISJOINT POSITIVE 1-IN-3SAT
INSTANCE: Set U of variables, collection of subsets $\mathcal{C}$ of U such that $|c| = 3$ for each subset $c \in \mathcal{C}$ and each pair of subsets $c, d \in \mathcal{C}, c \neq d$ satisfies $|c \cap d| \leq 1$.
QUESTION: Is there a truth assignment $\beta : U \to \{0,1\}$ such that each subset in $\mathcal{C}$ has exactly one true variable?

POSITIVE 1-IN-3SAT, which does not have the condition $|c \cap d| \leq 1$, is already known to be NP-complete [7, p. 259]. We show that it remains NP-complete on the requirement of almost disjointness.

Lemma 4 ALMOST DISJOINT POSITIVE 1-IN-3SAT *is* NP*-complete.*

Proof. (Omitted.) □

Theorem 5 CONSISTENCY FOR $\mathcal{B}$ WITH COINCIDENCE 1 AND HEAVINESS 4 *is* NP*-complete.*

Proof. (Outline.) The proof is by reduction from ALMOST DISJOINT POSITIVE 1-IN-3SAT. Let $(U, \mathcal{C})$ be an instance of the latter where $U = \{u_1, \ldots, u_n\}$. We define a set of examples S where $\mathrm{dom}(S) \subseteq \{0,1\}^{4n+2}$. For every $V \subseteq U$ let $\overline{1}_V$ denote the element $x \in \{0,1\}^n$ where $x_i = 1$ iff $u_i \in V$. We also write $\overline{0}$ instead of $\overline{1}_\emptyset$. The components $1, \ldots, 4n+2$ are partitionend into six blocks, the first four comprising n components each, the last two consisting both of a single component. For each variable $u_i \in U$ we define four examples

$$\begin{aligned}(\overline{1}_{\{u_i\}}, \overline{1}_{\{u_i\}}, \overline{0}, \overline{0}, 0, 0), (\overline{0}, \overline{0}, \overline{1}_{\{u_i\}}, \overline{1}_{\{u_i\}}, 0, 0) &\in \mathrm{neg}(S)\\ (\overline{1}_{\{u_i\}}, \overline{0}, \overline{0}, \overline{1}_{\{u_i\}}, 1, 0), (\overline{0}, \overline{1}_{\{u_i\}}, \overline{1}_{\{u_i\}}, \overline{0}, 1, 0) &\in \mathrm{pos}(S).\end{aligned}$$

Each subset $c \in \mathcal{C}$ is represented by two examples

$$(\overline{1}_c, \overline{0}, \overline{0}, \overline{0}, 0, 1), (\overline{0}, \overline{1}_c, \overline{0}, \overline{0}, 0, 0) \in \mathrm{pos}(S).$$

Finally, we define three examples

$$\begin{aligned} (\overline{0},\overline{0},\overline{0},\overline{0},1,1) &\in \mathrm{pos}(S) \\ (\overline{0},\overline{0},\overline{0},\overline{0},1,0),(\overline{0},\overline{0},\overline{0},\overline{0},0,1) &\in \mathrm{neg}(S). \end{aligned}$$

Obviously, $\zeta(S) = 1$, $\eta(S) = 4$, and S can be computed from $(U, \mathcal{C})$ in polynomial time. It can be shown that $\mathcal{C}$ has a 1-IN-3SAT assignment if and only if there exists a function $f : \{0,1\}^{4n+2} \to \{0,1\}$ in $\mathcal{B}$ that is consistent with S. □

Theorem 6 MAXIMUM CONSISTENCY FOR $\mathcal{B}$ WITH COINCIDENCE 1 AND HEAVINESS 2 *is* NP*-complete.*

Proof. (Outline.) The proof is an adaptation of a proof by Höffgen and Simon [10, Theorem 3.1]. It turns out that a modification of their construction is sufficient for our purpose. The reduction is from VERTEX COVER [7, p. 190]. Let $(V, E), K$ be an instance of the latter where $V = \{v_1, \ldots, v_n\}$. The set of examples S where $\mathrm{dom}(S) \subseteq \{0,1\}^{2n}$ is constructed as follows. For each $v_i \in V$ we define a vertex example $(\overline{1}_{\{v_i\}}, \overline{1}_{\{v_i\}}) \in \mathrm{neg}(S)$. For each $e \in E$ we define two edge examples $(\overline{1}_e, \overline{0}), (\overline{0}, \overline{1}_e) \in \mathrm{pos}(S)$. We omit the rest of the proof. □

Höffgen and Simon have investigated single neurons with arbitrary weights. The following statement is an immediate consequence of their proof [10, Theorem 3.1].

Corollary 7 MAXIMUM CONSISTENCY FOR $\mathcal{L}$ WITH COINCIDENCE 1 AND HEAVINESS 2 *is* NP*-complete.*

4 Learning problems solvable in linear time

In this section we show that all (maximum) consistency problems complementary to those of the previous section have linear-time algorithms on a random access machine. Thus, we not only obtain a complete characterization of the complexity of consistency problems for $\mathcal{B}$ and $\mathcal{L}$ with bounded coincidence and heaviness, but also the striking result that, with respect to a random access machine, the problem classes solely consist of elements of the highest and—disregarding sublinear time—lowest complexity in NP.

Lemma 8 CONSISTENCY FOR $\mathcal{B}$ WITH COINCIDENCE 0 *is solvable in linear time.*

Proof. (Outline.) If $\zeta(S) = 0$ then the examples do not overlap. Thus, we can assign weights for each example independently, in particular in linear time. □

Theorem 9 CONSISTENCY FOR $\mathcal{B}$ WITH HEAVINESS 3 *is solvable in linear time.*

Proof. (Outline.) The algorithm has three parts corresponding to the following partition of values the threshold may assume: $t \leq 1, t \geq 3$, and $t = 2$. In case $t \leq 1$ we let $t = 0$ if $\mathrm{neg}(S) = \emptyset$, otherwise we let $t = 1$ and ensure that $w \cdot x = 0$ for all $x \in \mathrm{neg}(S)$. In case $t \geq 3$ we let $t = 3$ and ensure that $w \cdot x = 3$ for all $x \in \mathrm{pos}(S)$. (Note that t need not be larger than 3 because $\eta(S) \leq 3$.) To handle the case $t = 2$ we make a reduction to 2-SATISFIABILITY. Figure 1 shows the algorithm. 2-SATISFIABILITY can be decided in linear time [3, 6]. The reduction and the first two parts ($t \leq 1, t \geq 3$) can easily be seen to run in linear time. □

Algorithm
Input: Set S of examples with $\eta(S) \leq 3$
Output: Set $\mathcal{C}$ of clauses of size 2 over the variables $\{w_1, \ldots, w_n\}$ such that there is a function f in $\mathcal{B}$ with threshold 2 consistent with S iff $\mathcal{C}$ has a satisfying assignment.
(1) Initialize $\mathcal{C} := \emptyset$;
(2) For all $x \in \mathrm{dom}(S)$
let $I = \{i \mid x_i = 1\}$;
(2.1) if $|I| = 3$ let $\{j, k, l\} = I$ and
$\mathcal{C} := \mathcal{C} \cup \{ \{w_j, w_k\}, \{w_j, w_l\}, \{w_k, w_l\} \}$ if $x \in \mathrm{pos}(S)$
$\mathcal{C} := \mathcal{C} \cup \{ \{\overline{w}_j, \overline{w}_k\}, \{\overline{w}_j, \overline{w}_l\}, \{\overline{w}_k, \overline{w}_l\} \}$ if $x \in \mathrm{neg}(S)$
(2.2) if $|I| = 2$ let $\{j, k\} = I$ and
$\mathcal{C} := \mathcal{C} \cup \{ \{w_j\}, \{w_k\} \}$ if $x \in \mathrm{pos}(S)$
$\mathcal{C} := \mathcal{C} \cup \{ \{\overline{w}_j, \overline{w}_k\} \}$ if $x \in \mathrm{neg}(S)$
(2.3) if $|I| = 1$ let $\{j\} = I$ and
$\mathcal{C} := \mathcal{C} \cup \{ \{w_j\}, \{\overline{w}_j\} \}$ if $x \in \mathrm{pos}(S)$

Figure 1: Reduction used in the proof of Theorem 9

Theorem 10 MAXIMUM CONSISTENCY FOR $\mathcal{B}$ WITH COINCIDENCE 0 *as well as* MAXIMUM CONSISTENCY FOR $\mathcal{B}$ WITH HEAVINESS 1 *is solvable in linear time.*

Proof. (Omitted.) □

Corollary 11 MAXIMUM CONSISTENCY FOR $\mathcal{L}$ WITH COINCIDENCE 0 *as well as* MAXIMUM CONSISTENCY FOR $\mathcal{L}$ WITH HEAVINESS 1 *is solvable in linear time.*

5 Concluding remarks

We have aimed to investigate sufficient and necessary conditions for the existence of efficient learning algorithms. The approach was to introduce criteria for the complexity of example sets to define simpler problem instances for known hard problems. This is different from a recently proposed polynomial-time learning algorithm by Redding *et al.* [17] who make use of higher order networks of varying size and node fan-in. We confined ourselves to fixed architectures showing that extensive knowledge about the examples one has to deal with can lead to highly specialized and fast learning algorithms. The line taken here can be considered a contrast to that of the backpropagation procedure which is a simple universal but slow algorithm.

With respect to considered architectures and restrictions we have shown exact dividing lines between polynomial-time solvable and NP-complete learning problems. However, if a given set of examples does not fall into one of our feasible restrictions this does not necessarily imply that training must be slow. There may exist further properties waiting for discovery that prevent learning from being intractable.

References

[1] E. Amaldi. On the complexity of training Perceptrons. In T. Kohonen, K. Mäkisara, O. Simula, and J. Kangas, editors, *Artificial Neural Networks*, pages 55–60. Elsevier (North-Holland), 1991.

[2] M. Anthony and N. Biggs. *Computational Learning Theory.* Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, Cambridge, 1992.

[3] B. Aspvall, M. F. Plass, and R. E. Tarjan. A linear-time algorithm for testing the truth of certain quantified Boolean formulas. *Information Processing Letters*, 8:121–123, 1979.

[4] A. L. Blum and R. L. Rivest. Training a 3-node neural network is NP-complete. *Neural Networks*, 5:117–127, 1992.

[5] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM*, 36:929–965, 1989.

[6] S. Even, A. Itai, and A. Shamir. On the complexity of timetable and multicommodity flow problems. *SIAM Journal on Computing*, 5:691–703, 1976.

[7] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, San Francisco, 1979.

[8] S. Haykin. *Neural Networks: A Comprehensive Foundation.* Macmillan, New York, 1994.

[9] J. Hertz, A. Krogh, and R. G. Palmer. *Introduction to the Theory of Neural Computation*, volume I of *Santa Fe Institute studies in the sciences of complexity.* Addison-Wesley Publishing Company, Redwood City, CA, 1991.

[10] K.-U. Höffgen and H.-U. Simon. Robust trainability of single neurons. In *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory*, pages 428–439. ACM Press, New York, 1992.

[11] J. S. Judd. *Neural Network Design and the Complexity of Learning.* The MIT Press, Cambridge, CA, 1990.

[12] H. Karloff. *Linear Programming.* Birkhäuser, Boston, 1991.

[13] J. F. Kolen and J. B. Pollack. Backpropagation is sensitive to initial conditions. *Complex Systems*, 4:269–280, 1990.

[14] J.-H. Lin and J. S. Vitter. Complexity results on learning by neural nets. *Machine Learning*, 6:211–230, 1991.

[15] B. K. Natarajan. *Machine Learning: A Theoretical Approach.* Morgan Kaufmann, San Mateo, CA, 1991.

[16] L. Pitt and L. G. Valiant. Computational limitations on learning from examples. *Journal of the ACM*, 35:965–984, 1988.

[17] N. J. Redding, A. Kowalczyk, and T. Downs. Constructive higher-order network algorithm that is polynomial time. *Neural Networks*, 6:997–1010, 1993.

[18] M. Schmitt. On the complexity of consistency problems for neurons with binary weights. Ulmer Informatik-Berichte 94-01, Universität Ulm, January 1994.

[19] J. Šíma. Loading deep networks is hard. *Neural Computation*, 6:842–850, 1994.