Exploiting the Link Between Learning Theory and Algorithmic Discrete Mathematics

Don Hush
CIC-3, MS B265
dhush@lanl.gov

February 1, 2000

A Proposal to the Los Alamos Applied Mathematical Sciences Program

1 Introduction

Many problems of national concern are not easily solved using a *first principles* model. Relevant examples include Information Security (e.g. computer network intrusion detection, or automatic document classification), and Intelligent Surveillance (e.g. the detection and characterization of a specific phenomenon using remote sensor data). In such cases it has become popular to use the computer to *learn* the correct model based on *examples* of its behavior. The notion of "learning from examples" has an intuitive appeal that stems from the analogy to the human learning experience. This analogy suggests that such a task can be easily accomplished with a small number of examples. But such analogies are misleading in that the learning problem appears to be more difficult for computers. This brings into question fundamental issues related to the tractability and efficiency of computer learning, and its potential for success in the real world. We propose a rigorous study of these issues using conventional models of computing.

Machine Learning (ML) is the science of building predictors from randomly generated data while accounting for the computational complexity of the learning algorithm and the predictor's performance on future data. The seemingly unrelated field of Algorithmic Discrete Mathematics (ADM) is concerned with the design of efficient algorithms for discrete mathematical problems such as Traveling Salesman, Matching, Graph Coloring, Partitioning, etc. Many of these ADM problems are intractable in the worst case setting (assuming $P \neq NP$), but some appear to have a large fraction of instances that can be solved efficiently. This has motivated a probabilistic study of these problems. For example Frieze and Reed [5] have constructed a polynomial-time algorithm that with high probability solves the Hamiltonian cycle problem on random graphs with edge probability $1/2$. They also show that a certain branch and bound algorithm for the Knapsack problem takes super-polynomial time with high probability. For recent surveys of this field see Alon, et.al. [1] and Habib, et.al. [6].

The incorporation of probability into the ADM framework creates a similarity between the development and analysis of algorithms for Machine Learning and Algorithmic Discrete Mathematics which we expand upon below. Historically the ML community has placed more emphasis on predictor performance than computational complexity and thus has produced powerful probabilistic tools for bounding the prediction error. Although many of these same probabilistic tools can be used to analyze the computational complexity of learning algorithms, their application in this domain is still quite primitive. In contrast, the strong emphasis on computational complexity in the ADM community has resulted in a sophisticated use of the probabilistic method for algorithm analysis which should be useful in advancing the computational complexity component of ML. At the same time the probabilistic method employed by the ADM community could be significantly enriched by incorporating the advanced probabilistic tools from ML. Thus, it appears that ML and ADM can benefit from each other's strengths.

Because of its de-emphasis on computational complexity, ML has fallen short of its goal of completely addressing the concerns of the practitioner. In this proposal we revive the emphasis on computational complexity in an attempt to remedy this situation. Likewise we propose to advance the state-of-the art in ADM by applying advanced probabilistic tools from ML to both the analysis of approximate solutions and algorithm complexity.

2 A Simple Connection Between ML and ADM

Here we demonstrate a simple connection between ML and ADM by showing that there is an isomorphism between the landmark problems of *linear classifier design* in ML and *satisfiability* in ADM. Having done so allows the exchange of complexity and performance results between the ML and ADM communities. First we introduce some terminology.

A linear threshold function (LTF) is a binary function of the form $y = H(w \cdot x + b)$, with parameters w and b, where H is the Heaviside function defined by $H(z) = 0$ for $z < 0$ and $H(z) = 1$ otherwise. For simplicity we restrict the components of x

and w to be binary $\{-1, 1\}$ and b to be integral. In the ML framework the LTF plays the role of a linear classifier for a two-class pattern recognition problem where x is a pattern vector, and $\{w, b\}$ are the parameters of the classifier. In the ADM framework the LTF plays the role of a logical expression, where the components of x are the literals and $\{w, b\}$ are the parameters of the logic function. Now we define the ML problem LT-LOADING and the ADM problem LT-k-SAT and show that they are isomorphic.

LT-k-SAT: *Given k LTFs $y = H(w_j \cdot x + b_j)$, $j = 1, ..., k$ with parameters w_j, b_j, does there exist an assignment for x for which $y = 1$ for all j?*

Note that this problem can be formulated as a satisfiability problem for the conjunction of LTFs,

$$H(w_1 \cdot x + b_1) \wedge H(w_2 \cdot x + b_2) \wedge ... \wedge H(w_k \cdot x + b_k)$$

LT-LOADING: *Given a set $\{x_j, y_j\}, j = 1, 2, ..., k$ does there exist an LTF with parameters $\{w, b\}$ such that all k points are correctly classified, that is $y_j = H(w \cdot x_j + b)$ for all j?*

Note that this problem can be formulated as a satisfiability problem for the boolean formula

$$H((2y_1 - 1)(w \cdot x_1 + b)) \wedge H((2y_2 - 1)(w \cdot x_2 + b)) \wedge ... \wedge H((2y_k - 1)(w \cdot x_k + b))$$

Letting $\acute{x}_i = (2y_i - 1)x_i$ and $\acute{b}_i = (2y_i - 1)b$ this becomes

$$H(w \cdot \acute{x}_1 + \acute{b}) \wedge H(w \cdot \acute{x}_2 + \acute{b}) \wedge ... \wedge H(w \cdot \acute{x}_k + \acute{b})$$

so that mapping $\acute{x}_i$ to w_i, $\acute{b}$ to b_i and w to x completes the isomorphism between LT-LOADING and LT-k-SAT.

This mapping also provides isomorphisms for both the *search* and *optimization* versions of these problems. In the search problem we ask for a solution vector for the yes instances of the decision problem. In the optimization problem we ask for a solution vector that maximizes the number of LTFs satisfied in the conjunction. In ML the search and optimization problems correspond to the landmark problem of linear classifier design for separable and non-separable data respectively.

3 Algorithm Development Framework

The linear classifier design problem just described is NP-Complete [9], meaning that there are instances that cannot be solved in polynomial time (assuming $P \neq NP$).

However there may be variations that admit polynomial-time solutions. For example hard instances may be rare, so that by placing a distribution on the space of instances we may be able to efficiently produce exact solutions with high probability. Note that in ML the distribution is determined by nature whereas in ADM it is specified by the analyst in a manner that is consistent with the problem domain. Alternatively this problem may lend itself to efficient solutions through the use of randomized algorithms (i.e. it may live in RP). Randomized algorithms and randomized instances are two different ways of introducing randomization that may also be considered simultaneously. On the other hand it may be that relaxing the requirement of an exact solution may make the problem tractable. For example it is known that relaxing the binary constraints on the LTF parameters leads to a tractable problem (linear programming) when the data is linearly separable [9]. By mapping this solution to one with binary coefficients we obtain an approximate solution. Thus we have converted an NP-Complete problem to an approximation that can be solved in $O(dk(d+k)^4)$ time where k is the number of pattern vectors and d is their dimension. In addition, there may be instances for which an approximate solution cannot be achieved in polynomial time leading us to consider a randomized treatment of approximate solutions.

A taxonomy of the various scientific regimes discussed above is illustrated in Figure 1. We would like to use this taxonomy to identify problems where the probabilistic tools for bounding performance from ML can be applied to ADM, and the probabilistic methods of algorithm analysis from ADM can be applied to ML. To do so will require the development of a formal model for individual nodes in this taxonomy. One example is Valiant's Probably Approximately Correct (PAC) model of ML [10] which requires that the computational complexity be controlled as a function the accuracy of approximation and the probability of encountering a non-representative instance.

4 Proposed Work

We have described a general framework for exploiting the link between ML and ADM. We now describe two specific examples of problems that we will investigate.

1. Linear classifier design with nonseparable data is known to be NP-Hard, but may be tractable under the appropriate distributional assumptions. For example, Bylander has positive results for on-line classification error in terms of a measure of separability of the data [3, 4]. We propose to combine these results with techniques from ADM to obtain an efficient learning algorithm with a well-defined stopping criterion and good performance. In doing so we will have

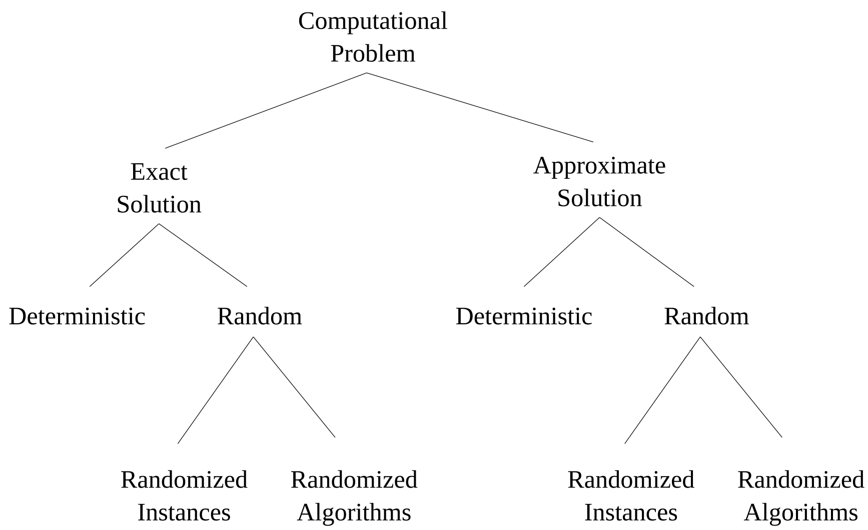

Figure 1: Taxonomy of regimes for the study of ADM

resolved a landmark problem in Machine Learning and produced a computationally effective algorithm for use by the practitioner.

2. ADM often provides positive results for the average run-time, but probabilistic bounds on run-time are rare largely because the corresponding algorithms are not amenable to existing concentration of measure theory. We propose to remedy this situation by further investigating general concentration bounds like those in Rhee and Talagrand [8], Boucheron, et.al. [2], and Hush, et.al. [7]. In addition we propose to investigate concentration bounds for specific problems in ADM.

5 Funding Request

We request the standard funding level of $100K per year for four years.

References

[1] Alon, N., Spencer, J. H., and Erdös, P., *The Probabilistic Method*, John Wiley, New York, 1992.

[2] Boucheron, S., Lugosi, G., and Massart, P., A sharp inequality with applications, preprint, 1999.

[3] Bylander, T., Polynomial learnability of linear threshold approximations, *Proceedings of the Sixth Annual ACM Conference on Computational Learning*(1993), 297–302

[4] Bylander, T., Learning noisy linear threshold functions, *Preprint* (1998).

[5] Frieze, A. M. and Reed, B., Probabilistic Analysis of Algorithms, *Probabilistic Methods for Algorithmic Discrete Mathematics* Habib, M., McDiarmid, C., Ramirez-Alfonsin, J., Reed, B., Eds., pp. 36–92, Springer-Verlag, Berlin, 1998.

[6] Habib, M., McDiarmid, C., Ramirez-Alfonsin, J., Reed, B., Eds., *Probabilistic Methods for Algorithmic Discrete Mathematics* Springer-Verlag, Berlin, 1998.

[7] Hush, D., Scovel, C., A new proof of concentration of Rademacher statistics, submitted to *Annals of Probability*, 1999.

[8] Rhee, W. T., and Talagrand, M., A sharp deviation for the stochastic travelling salesman problem, *Ann. Probab* **17**(1989), 1–8.

[9] Siu, K.-Y., Roychowdhury, V., and Kailath, T., *Discrete Neural Computation: A Theoretical Foundation*, Prentice–Hall, Englewood Cliffs, NJ, 1995.

[10] Valiant, L.G., A theory of the learnable, *Communications of the Association for Computing Machinery* **27:11**(1984), 1134–1142.