Machine Teaching as Search

Scott Alfeld[*] and **Xiaojin Zhu**[*] and **Paul Barford**[*†]
[*]Department of Computer Sciences, University of Wisconsin – Madison, Madison WI 53706, USA
[†]comScore, Inc., 11950 Democracy Drive, Suite 600, Reston, VA 20190, USA.
{salfeld, jerryzhu, pb}@cs.wisc.edu

Abstract

Machine teaching (MT) studies the task of designing a training set. Specifically, given a learner (e.g., an artificial neural network or a human) and a target model, a teacher aims to create a training set which results in the target model being learned. MT applications include optimal education design for human learners and computer security where adversaries aim to attack learning-based systems. In this work, we formulate pool-based MT as a state space search problem. We discuss the properties and challenges of the resulting problem and highlight opportunities for novel search techniques. In our preliminary study we use a beam search approach, and find that training and evaluating empirical risk of models dominate the run time of the search. Toward the goal of better search techniques for future work, we develop optimizations ranging from implementation details for specific learners to algorithm changes applicable to general blackbox learners. We conclude with a discussion of open problems and research directions.

Introduction

Machine Teaching (MT) studies the task of creating an training set for a given learner and target model. For a survey of the area, see (Zhu 2015). It began with a theoretical investigation of the so-called *teaching dimension* (Goldman and Kearns 1995; Shinohara and Miyano 1991), and may be thought of as the inverse problem of machine learning. It has been gaining growing interest in part due to applications in optimal education design (Patil et al. 2014), where the goal is to create material (e.g., lesson plans or homework exercises) for human learners. In addition, a driving application has been security (Mei and Zhu 2015; Barreno et al. 2006; 2010; Alfeld, Zhu, and Barford 2016; Huang et al. 2011) where an adversary aims to corrupt a learning system for her own gains. For example, an attacker sends carefully crafted spam emails to her own account and marks them as benign so as to poison a machine-learning based spam filter (Nelson et al. 2009; Lowd and Meek 2005). In this work, we introduce Machine Teaching as a novel application of combinatorial search.

MT considers two scenarios: *constructive*, where a teacher can freely synthesize training items, and *pool-based*. In this work we focus on the pool-based setting, where we are given a finite candidate pool $C = \{(x, y)\}_i$ from which to select training items. Given a cost function and budget s, we denote the set of all possible (multi)sets of cost at most s as $\mathcal{D}$. In this work we let the cost function be the size of the teaching set: $\mathcal{D} = \{D \subseteq C \text{ s.t. } |D| \leq s\}$. Many applications (e.g., computer vision tasks) fall into the pool-based setting, as a teacher cannot easily synthesize arbitrary examples.

Given a hypothesis space $\mathcal{H}$, a learner $\mathcal{A} : \mathcal{D} \mapsto \mathcal{H}$, and *teaching risk* function $\ell : \mathcal{H} \to \mathbb{R}_+$, we aim to find a teaching set $D \in \mathcal{D}$ for which $\ell(\mathcal{A}[D])$ is small. Here $\mathcal{A}[D]$ denotes the resulting hypothesis of $\mathcal{A}$ being trained on D, and the function $\ell(\cdot)$ may be thought of as a distance function to some desired target hypothesis h^*. For example $\ell(h) = \mathbb{P}_x(h(x) \neq h^*(x))$ or alternatively the empirical loss on some test set $T = \{(x, y)\}_i$: $\ell(h) = \sum_{i \in T} \mathbb{1}(h(x_i) \neq y_i)$. Our goal is therefore to find an element of

$$\arg\min_{D \in \mathcal{D}} \ell(\mathcal{A}[D]) \tag{1}$$

We note a special case of MT, where a teacher aims to teach a model in alignment with the candidate pool – that is, $T = C$. In such settings, the task of machine teaching is akin to that of training set reduction (Wilson and Martinez 2000). Namely, given a (large) training set, select a (small) subset which yields an equivalent model. Our setting is more general, as the target model need not be one which performs well on the original data (e.g., in adversarial settings).

Indeed, our setting is very flexible, as we illustrate with an example. In both optimal education design and adversarial learning, teaching a known model is not the only task at hand. Often the details of the learner's algorithm are unknown, and must be inferred through e.g., probing. Suppose we have two proposed learners $\mathcal{A}_1$ and $\mathcal{A}_2$, and we want to figure out which one our black-box learner $\mathcal{A}$ is. One way to accomplish this is to find a training set D_{tr} and test set D_{te} such that

$$\text{Hamming-Distance}(\mathcal{A}_1[D_{\text{tr}}](D_{\text{te}}), \mathcal{A}_2[D_{\text{tr}}](D_{\text{te}})) \tag{2}$$

is maximized. That is, the two learners are distinguished by training them on D_{tr} followed by having them each predict D_{te}. If D_{te} is fixed, we can solve this problem defining our loss function as the number of items (in D_{te}) on which $\mathcal{A}_1$

and $\mathcal{A}_2$ agree. This yields the following optimization problem

$$\arg\min_{D\in\mathcal{D}} \sum_{x\in D_{\text{te}}} \mathbb{1}(\mathcal{A}_1[D](x) = \mathcal{A}_2[D](x)) \qquad (3)$$

We leave the task of simultaneously minimizing over D and D_{te} as one avenue of future work.

Structure of the State Space

By defining actions as adding items, rather than allowing removing items as well, we impose a partial ordering on states. That is, nodes at level d correspond to states with exactly d items. Because of this, we do not need to maintain a closed list, but instead only check membership in the frontier when adding successors. This applies to any learner, and additional enhancements may be available based on properties of machine learning algorithms, as described below.

In addition, in some cases the state space forms a tree instead of a graph. This occurs when the learner's behavior depends on the *order* of items in the training set e.g., online learners. States then correspond with training *sequences* rather than states, and thus tree (rather than graph) search methods may be used.

Fast Node Evaluation

For some learners, there is no clear way to update from $\mathcal{A}[D]$ to $\mathcal{A}[D \cup (x, y)]$ without completely retraining. In such cases, training models takes a considerable portion of the runtime.

Some learners, however, such as k-Nearest Neighbors (kNN) and Support Vector Machines (Boser, Guyon, and Vapnik 1992) have certain local structure when updating. For example a linear SVM will not change its decision boundary if the new item is outside the margin, and the effect of a single item on kNN's decision boundary will be localized. We use this to retrain models only when necessary, reducing total runtime.

In addition, for various learners sequential learning is possible. That is, computing $\mathcal{A}[D\cup(x, y)]$ is faster if we already know $\mathcal{A}[D]$. A classical example is the Perceptron learner (Rosenblatt 1958), which learns in an online fashion and can quickly update its model given a new training item.

Summary and Future Work

In this extended abstract we have phrased machine teaching in the pool-based setting as a combinatorial search problem. Based on our initial investigations, there is a rich set of opportunities for search-based advancements to this problem. We outline two directions of future work beyond improving on our preliminary optimizations discussed above.

(i) Leveraging the smoothness of learners. Training items often lie in $\mathbb{R}^n$ and a learner shows certain smoothness properties. That is, if $x_i \approx x_j$ and $y_i \approx y_j$, then $\mathcal{A}[D \cup \{(x_i, y_i)\}] \approx \mathcal{A}[D \cup \{(x_j, y_j)\}]$. Exploiting this may yield heuristics useful in pruning the search space.

(ii) Reusing loss computations. In our experiments with beam search, we let $\ell(\cdot)$ be defined as empirical risk on the candidate pool:

$$\ell(\mathcal{A}[D]) \triangleq \sum_{i=1}^{\|C\|} \mathbb{1}(\mathcal{A}[D](x_i) = y_i) \qquad (4)$$

We discover that the runtime is, in general, dominated by evaluating empirical risk. Due to the smoothness properties of learners mentioned above, many cycles are wasted re-evaluating empirical risk across large subsets of the candidate pool. Either (quickly) approximating empirical risk, or utilizing previous calculations to speed up exact computation may yield more efficient search.

Acknowledgments

This material is based upon work supported in part by NSF grant IIS 0953219, the University of Wisconsin – Madison Graduate School with funding from the Wisconsin Alumni Research Foundation, and by the Laboratory for Telecommunication Sciences, DHS grant BAA 11-01 and AFRL grant FA8750- 12-2-0328. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of LTS, DHS or AFRL.

References

Alfeld, S.; Zhu, X.; and Barford, P. 2016. Data poisoning attacks against autoregressive models. In *AAAI*.

Barreno, M.; Nelson, B.; Sears, R.; Joseph, A. D.; and Tygar, J. D. 2006. Can machine learning be secure? In *CCS*.

Barreno, M.; Nelson, B.; Joseph, A. D.; and Tygar, J. 2010. The security of machine learning. *Machine Learning*.

Boser, B. E.; Guyon, I. M.; and Vapnik, V. N. 1992. A training algorithm for optimal margin classifiers. In *COLT*.

Goldman, S. A., and Kearns, M. J. 1995. On the complexity of teaching. *Journal of Computer and System Sciences*.

Huang, L.; Joseph, A. D.; Nelson, B.; Rubinstein, B. I.; and Tygar, J. 2011. Adversarial machine learning. In *AISEC*.

Lowd, D., and Meek, C. 2005. Good word attacks on statistical spam filters. In *CEAS*.

Mei, S., and Zhu, X. 2015. Using machine teaching to identify optimal training-set attacks on machine learners. In *AAAI*.

Nelson, B.; Barreno, M.; Chi, F. J.; Joseph, A. D.; Rubinstein, B. I.; Saini, U.; Sutton, C.; Tygar, J.; and Xia, K. 2009. Misleading learners: Co-opting your spam filter. In *Machine learning in cyber trust*. Springer.

Patil, K. R.; Zhu, X.; Kopeć, Ł.; and Love, B. C. 2014. Optimal teaching for limited-capacity human learners. In *NIPS*.

Rosenblatt, F. 1958. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*.

Shinohara, A., and Miyano, S. 1991. Teachability in computational learning. *New Generation Computing*.

Wilson, D. R., and Martinez, T. R. 2000. Reduction techniques for instance-based learning algorithms. *Machine learning*.

Zhu, X. 2015. Machine teaching: An inverse problem to machine learning and an approach toward optimal education. In *AAAI*.