Learning DFA from Simple Examples

Rajesh Parekh and Vasant Honavar*

Artificial Intelligence Research Group
Department of Computer Science
Iowa State University
Ames IA 50011-1040. U.S.A.
{parekh|honavar}@cs.iastate.edu

Abstract. We present a framework for learning DFA from *simple* examples. We show that efficient PAC learning of DFA is possible if the class of distributions is restricted to *simple* distributions where a teacher might choose examples based on the knowledge of the target concept. This answers an interesting variant of an open research question posed in Pitt's seminal paper: *Are DFA's PAC-identifiable if examples are drawn from the uniform distribution, or some other known simple distribution?* Our approach uses the RPNI algorithm for learning DFA from labeled examples. In particular, we describe an efficient learning algorithm for exact learning of the target DFA with high probability when a bound on the number of states (N) of the target DFA is known in advance. When N is not known, we show how this algorithm can be used for efficient PAC learning of DFAs.

1 Introduction

The problem of learning the minimum state DFA that is consistent with a given sample has been actively studied for over two decades. DFAs are recognizers for *regular* languages which constitute one of the simplest class of languages in the formal language hierarchy [Chomsky, 1956; Hopcroft & Ullman, 1979]. An understanding of the issues and problems encountered in learning regular languages (or equivalently, identification of the corresponding DFA) are therefore likely to provide insights into the problem of learning more general classes of languages.

Exact learning of the target DFA from an arbitrary presentation of labeled examples is a hard problem [Gold, 1978]. Gold showed that the problem of identifying the minimum state DFA consistent with a presentation S comprising of a finite non-empty set of positive examples S^+ and possibly a finite non-empty set of negative examples S^- is NP-hard. Under the standard *complexity theoretic* assumption $P \neq NP$, Pitt and Warmuth showed that no polynomial time algorithm can be guaranteed to produce a DFA with at most $N^{(1-\epsilon)\log\log(N)}$ states

* This research was partially supported by grants from the National Science Foundation (IRI-9409580 and IRI-9643299) and the John Deere Foundation to Vasant Honavar.

from a set of labeled examples corresponding to a DFA with N states [Pitt & Warmuth, 1988].

Efficient learning algorithms for identification of DFA assume that additional information is provided to the learner. Trakhtenbrot and Barzdin described a polynomial time algorithm for constructing the smallest DFA consistent with a *complete labeled sample* i.e., a sample that includes all strings up to a particular length and the corresponding label that states whether the string is accepted by the target DFA or not [Trakhtenbrot & Barzdin, 1973]. Angluin showed that given a *live-complete* set of examples (that contains a representative string for each live state of the target DFA) and a knowledgeable teacher to answer *membership queries* it is possible to exactly learn the target DFA [Angluin, 1981]. In a later paper, Angluin relaxed the requirement of a live-complete set and has designed a polynomial time inference algorithm using both *membership* and *equivalence* queries [Angluin, 1987]. The RPNI algorithm is a framework for identifying a DFA consistent with a given sample S in polynomial time [Oncina & García, 1992]. Further, if S is a superset of a characteristic set (see section 2.1) for the target DFA then the DFA output by the RPNI algorithm is guaranteed to be equivalent to the target [Oncina & García, 1992; Dupont, 1996b].

Pitt surveyed several approaches for *approximate* identification of DFA [Pitt, 1989]. Valiant's distribution-independent model of learning (also called the PAC model) [Valiant, 1984] is widely used for learning several different concept classes approximately. When adapted to the problem of learning DFA, the goal of a PAC learning algorithm is to obtain in polynomial time, with high probability, a DFA that is a good approximation of the target DFA. Even approximate learnability of DFA was proven to be a hard problem. Pitt and Warmuth showed that the problem of polynomially approximate predictability of the class of DFA is hard [Pitt & Warmuth, 1989]. They made use of *prediction preserving reductions* to show that if DFAs are polynomially approximately predictable then so are other known hard to predict concept classes such as *boolean formulas*. Further, under certain *cryptographic assumptions*, Kearns and Valiant showed that an efficient algorithm for learning DFA would entail efficient algorithms for solving several known hard problems such as breaking the *RSA* cryptosystem, factoring *Blum* integers, and detecting *quadratic residues* [Kearns & Valiant, 1989].

The PAC model's requirement of learnability under all conceivable distributions is often too stringent for practical learning scenarios. Pitt's paper identified the following open research problem: *Are DFA's PAC-identifiable if examples are drawn from the uniform distribution, or some other known simple distribution?* [Pitt, 1989]. Using a variant of Trakhtenbrot and Barzdin's algorithm, Lang empirically demonstrated that random DFAs are approximately learnable from a sparse uniform sample [Lang, 1992]. However, exact identification of the target DFA was not possible even in the average case with a randomly drawn training sample. Several concept classes are efficiently PAC learnable under restricted classes of distributions while their learnability under the distribution free model is not known [Li & Vitányi, 1991]. Li and Vitányi proposed a model for PAC learning with *simple* examples wherein the examples are drawn according to the *Solomonoff-Levin universal distribution*. They showed that learnability under the universal distribution implies learnability under a broad class of simple distributions. Thus, this model is sufficiently general. Recently, this learning model has been extended to a framework where a teacher might choose examples based on the knowledge of the target concept [Denis *et al.*, 1996]. We show that under this extended framework of learning from simple examples (called the PACS model) it is possible to efficiently learn DFA thereby answering an interesting variant of Pitt's open research question in the affirmative.

The rest of this paper is organized as follows: Section 2 introduces the necessary definitions and notation. Section 3 summarizes the *RPNI* algorithm. Section 4 describes the learning of DFA with simple examples and section 5 concludes with a discussion of several interesting avenues that merit further investigation.

2 Preliminaries

Let Σ be a finite set of symbols called the *alphabet*; Σ^* be the set of strings over Σ; α, β, γ be strings in Σ^*; and $|\alpha|$ be the length of the string α. λ is a special string called the *null* string and has length 0. Given a string $\alpha = \beta\gamma$, β is the *prefix* of α and γ is the *suffix* of α. Let $Pr(\alpha)$ denote the set of all prefixes of α. A *language* L is a subset of Σ^*. The set $Pr(L) = \{\alpha \mid \alpha\beta \in L\}$ is the set of *prefixes* of the language and the set $L_\alpha = \{\beta \mid \alpha\beta \in L\}$ is the set of *tails* of α in L. The *standard order* of strings of the alphabet Σ is denoted by $<$. The standard enumeration of strings over $\Sigma = \{a, b\}$ is $\lambda, a, b, aa, ab, ba, bb, aaa, \ldots$ The set of *short prefixes* $S_p(L)$ of a language L is defined as $S_p(L) = \{\alpha \in Pr(L) \mid \nexists \beta \in \Sigma^*$ such that $L_\alpha = L_\beta$ and $\beta < \alpha\}$. The *kernel* $N(L)$ of a language L is defined as $N(L) = \{\lambda\} \cup \{\alpha a \mid \alpha \in S_p(L), a \in \Sigma, \alpha a \in Pr(L)\}$. Given two sets S_1 and S_2, let $S_1 \backslash S_2$ and $S_1 \oplus S_2$ denote the *set difference* and the *symmetric difference* respectively. Let ln and lg denote the log to the bases e and 2 respectively.

2.1 Finite Automata

A *deterministic* finite state automaton (DFA) is a quintuple $A = (Q, \delta, \Sigma, q_0, F)$ where, Q is a finite set of states, Σ is the finite alphabet, $q_0 \in Q$ is the start state, $F \subseteq Q$ is the set of accepting states, and δ is the transition function: $Q \times \Sigma \longrightarrow Q$. A state $d_0 \in Q$ such that $\forall a \in \Sigma,\ \delta(d_0, a) = d_0$ is called a *dead* state. The extension of δ to handle input strings is standard and is denoted by δ^*. The set of all strings accepted by A is its language, $L(A)$. The language accepted by a DFA is called a *regular language*. Fig. 1 shows the state transition diagram for a sample DFA. A *non-deterministic* finite automaton (NFA) is defined just like the DFA except that the transition function δ defines a mapping from $Q \times \Sigma \longrightarrow 2^Q$. In general, a finite state automaton (FSA) refers to either a DFA or a NFA.

Given any FSA (A'), there exists a minimum state DFA (also called the *canonical DFA*, A) such that $L(A) = L(A')$. We will assume that the target DFA being learned is a canonical DFA. Let N denote the number of states of A.

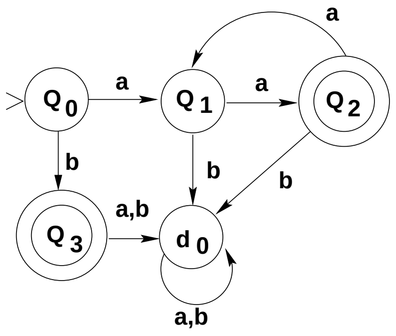

Fig. 1. Deterministic Finite State Automaton.

It can be shown that any canonical DFA has at most one dead state. A labeled example $(\alpha, c(\alpha))$ for A is such that $\alpha \in \Sigma^*$ and $c(\alpha) = +$ if $\alpha \in L(A)$ (i.e., α is a positive example) or $c(\alpha) = -$ if $\alpha \notin L(A)$ (i.e., α is a negative example). Let S^+ and S^- denote the set of *positive* and *negative* examples of A respectively. A is consistent with a *sample* $S = S^+ \cup S^-$ if it accepts all positive examples and rejects all negative examples.

A set S^+ is said to be *structurally complete* with respect to a DFA A if S^+ covers each transition of A (except the transitions associated with the dead state d_0) and uses every element of the set of final states of A as an accepting state [Pao & Carr, 1978; Parekh & Honavar, 1993; Dupont *et al.*, 1994]. It can be verified that the set $S^+ = \{b, aa, aaaa\}$ is structurally complete with respect to the DFA in Fig. 1. Given a set S^+, let $PTA(S^+)$ denote the *prefix tree acceptor* for S^+. $PTA(S^+)$ is a DFA that contains a path from the start state to an accepting state for each string in S^+ modulo common prefixes. Clearly, $L(PTA(S^+)) = S^+$. The PTA for the set S^+ (given above) is shown in Fig. 2.

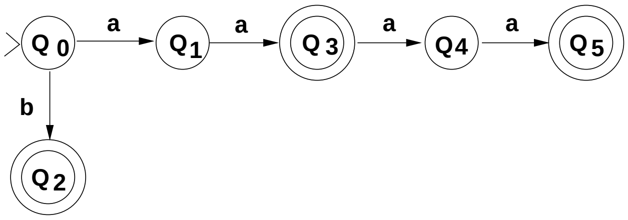

Fig. 2. Prefix Tree Automaton.

Given a FSA A and a partition π on the set of states Q of A (ignoring the dead state d_0 and its associated transitions), we define the *quotient automaton* $A_\pi = (Q_\pi, \delta_\pi, \Sigma, B(q_0, \pi), F_\pi)$ obtained by merging the states of A that belong to the same block of the partition π as follows: $Q_\pi = \{B(q, \pi) \mid q \in Q\}$ is the set of states with each state represented uniquely by the block $B(q, \pi)$ of the partition π that contains the state q, $F_\pi = \{B(q, \pi) \mid q \in F\}$ is the set of accepting states, and $\delta_\pi : Q_\pi \times \Sigma \longrightarrow 2^{Q_\pi}$ is the transition function such that $\forall B(q_i, \pi), B(q_j, \pi) \in Q_\pi$, $\forall a \in \Sigma$, $B(q_j, \pi) = \delta_\pi(B(q_i, \pi), a)$ *iff* $q_i, q_j \in Q$ and $q_j = \delta(q_i, a)$. Note that a quotient automaton of a DFA might be a NFA and vice-versa. For example, the quotient automaton corresponding to the partition $\pi = \{\{Q_0, Q_1\}, \{Q_2\}, \{Q_3\}\}$ of the set of states of the DFA in Fig. 1 is shown in Fig. 3.

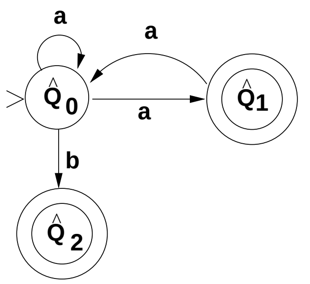

Fig. 3. Quotient Automaton.

The set of all derived automata obtained by systematically merging the states of A represents a *lattice* of FSA [Pao & Carr, 1978]. This lattice is ordered by the *grammar cover* relation $\preceq$. Given two partitions $\pi_i = \{B_1, B_2, \ldots, B_r\}$ and $\pi_j = \{B_1, B_2, \ldots, B_k\}$ of the states of A, we say that π_i covers π_j (written $\pi_j \preceq \pi_i$) if $r = k-1$ and for some $1 \leq l, m \leq k$, $\pi_i = \{\pi_j \backslash \{B_l, B_m\} \cup \{B_l \cup B_m\}\}$. The *transitive closure* of $\preceq$ is denoted by $\ll$. We say that $A_{\pi_j} \ll A_{\pi_i}$ *iff* $L(A_{\pi_j}) \subseteq L(A_{\pi_i})$. Given a canonical DFA A and a set S^+ that is structurally complete with respect to A, the lattice $\Omega(S^+)$ derived from $PTA(S^+)$ is guaranteed to contain A [Pao & Carr, 1978; Parekh & Honavar, 1993; Dupont *et al.*, 1994].

A sample $S = S^+ \cup S^-$ is said to be *characteristic* with respect to a regular language L (with a canonical acceptor A) if it satisfies the following two conditions [Oncina & García, 1992]:

- $\forall \alpha \in N(L)$, if $\alpha \in L$ then $\alpha \in S^+$ else $\exists \beta \in \Sigma^*$ such that $\alpha\beta \in S^+$.
- $\forall \alpha \in S_p(L), \forall \beta \in N(L)$, if $L_\alpha \neq L_\beta$ then $\exists \gamma \in \Sigma^*$ such that ($\alpha\gamma \in S^+$ and $\beta\gamma \in S^-$) or ($\beta\gamma \in S^+$ and $\alpha\gamma \in S^-$).

Intuitively, condition 1 implies structural completeness with respect to A and condition 2 implies that for any two distinct states of A there is a suffix γ that would correctly distinguish them. Given the language $L(A)$ corresponding to the DFA A in Fig. 1, the set of short prefixes is $S_p(L(A)) = \{\lambda, a, b, aa\}$ and the kernel is $N(L(A)) = \{\lambda, a, b, aa, aaa\}$. It can be easily verified that the set $S = S^+ \cup S^-$ where $S^+ = \{b, aa, aaaa\}$ and $S^- = \{\lambda, a, aaa, baa\}$ is a characteristic sample for $L(A)$.

2.2 Kolmogorov Complexity and the Universal Distribution

The *Kolmogorov Complexity* of an object x is a measure of the descriptional (or representational) complexity of x. Intuitively, the Kolmogorov complexity of an object with respect to a particular Turing Machine (M) is the length of the shortest description of the object on M. We will consider the *prefix* version of the Kolmogorov complexity denoted by K. Given a Turing machine implementing the partial recursive function $\phi : \{0,1\}^* \stackrel{partial}{\longrightarrow} \{0,1\}^*$, a program $\pi \in \{0,1\}^*$, and strings $\alpha, \beta \in \{0,1\}^*$, the Kolmogorov complexity of α relative to ϕ is defined as: $K_\phi(\alpha) = min\{|\pi| \mid \phi(\pi) = \alpha\}$ and the conditional Kolmogorov complexity of α given β relative to ϕ is defined as: $K_\phi(\alpha \mid \beta) = min\{|\pi| \mid \phi(\langle \pi, \beta \rangle) = \alpha\}$ where $\langle x, y \rangle$ is a standard pairing function[2]. Fixing a single universal Turing machine U we denote $K(\alpha) = K_U(\alpha)$ and $K(\alpha|\beta) = K_U(\alpha|\beta)$. It can be shown that $K(\alpha) \leq |\alpha| + \eta$ where η is a constant independent of α.

The *Solomonoff-Levin* universal distribution $\mathbf{m}$ is the universal enumerable probability distribution in that it multiplicatively dominates all enumerable probability distributions. It can be shown that $2^{-K(\alpha)} \leq \mathbf{m}(\alpha) \leq 2^{-K(\alpha)+\psi}$ where ψ is a constant independent of α. Thus, the universal distribution assigns higher probability to simple objects (i.e, objects with low Kolmogorov complexity). Given the knowledge of a string $r \in \Sigma^*$, the universal distribution $\mathbf{m}_r$, is defined as $\mathbf{m}_r(\alpha) = \lambda_r 2^{-K(\alpha|r)}$ where $\lambda_r \sum_{\alpha \in \Sigma^*} 2^{-K(\alpha|r)} = 1$ (i.e., $\lambda_r \geq 1$). The reader is referred to [Li & Vitányi, 1997] for a description of Kolmogorov complexity and related topics.

2.3 PAC Learning of DFA

Let $\mathcal{X}$ denote the *sample space* defined as the set of all strings Σ^*. Let $x \subseteq \mathcal{X}$ denote a *concept*. For our purpose, x is a *regular language*. We identify the concept with the corresponding canonical DFA and denote the class of all DFA as the *concept class* $\mathcal{C}$. The *representation* $\mathcal{R}$ that assigns a name to each DFA in $\mathcal{C}$ is defined as a function $\mathcal{R} : \mathcal{C} \longrightarrow \{0,1\}^*$ i.e., $\mathcal{R}$ represents the set of canonical encodings for the DFA in $\mathcal{C}$. Assume that there is an unknown and arbitrary but fixed distribution $\mathcal{D}$ according to which the examples of the target concept are drawn. In the context of learning DFA, $\mathcal{D}$ is restricted to a probability distribution on strings of Σ^* of length at most m.

Definition: [Pitt, 1989]
DFAs are PAC-identifiable *iff* there exists a (possibly randomized) algorithm $\mathcal{A}$ such that on input of any parameters ϵ and δ, for any DFA M of size N, for any number m, and for any probability distribution $\mathcal{D}$ on strings of Σ^* of length at most m, if $\mathcal{A}$ obtains labeled examples of M generated according to the distribution $\mathcal{D}$, then $\mathcal{A}$ produces a DFA M' with probability at least $1 - \delta$, the probability (with respect to distribution $\mathcal{D}$) of the set $\{\alpha \mid \alpha \in L(M) \oplus L(M')\}$

[2] Define $\langle x, y \rangle = bd(x)01y$ where bd is the bit doubling function defined as $bd(0) = 00$, $bd(1) = 11$, and $bd(ax) = aabd(x), a \in \{0,1\}$.

is at most ϵ. The run time of $\mathcal{A}$ (and hence the number of randomly generated examples obtained by $\mathcal{A}$) is required to be polynomial in N, m, $1/\epsilon$, $1/\delta$, and $|\Sigma|$.

3 The RPNI Algorithm

The *regular positive and negative inference* (RPNI) algorithm [Oncina & García, 1992] is a polynomial time algorithm for identifying a DFA consistent with a given sample $S = S^+ \cup S^-$. It can be shown that given a characteristic sample for the target DFA the algorithm is guaranteed to return a canonical representation of the target DFA [Oncina & García, 1992; Dupont, 1996b]. Our description of RPNI algorithm is based on the explanation given in [Dupont, 1996a]. The algorithm constructs a prefix tree acceptor $PTA(S^+)$ for the examples in S^+. The states of $PTA(S^+)$ are labeled based on the standard order of the set $Pr(S^+)$. An ordered search is performed in the space of partitions of the set of states of $PTA(S^+)$ under the control of S^-. The partition, π_0, corresponding to $PTA(S^+)$ itself is $\{\{0\},\{1\},\ldots,\{\overline{N}-1\}\}$ (where $\overline{N}$ is the number of states of $PTA(S^+)$).

Algorithm

Input: A sample $S = S^+ \cup S^-$
Output: A DFA compatible with S
begin
  $\pi = \pi_0 = \{\{0\},\{1\},\ldots,\{\overline{N}-1\}\}$
  $M = PTA(S^+)$
  for i = 1 **to** $\overline{N}-1$
    for j = 0 **to** $i-1$
      $\tilde{\pi} = \pi \backslash \{B(i,\pi), B(j,\pi)\} \cup \{B(i,\pi) \cup B(j,\pi)\}$
      $M_{\tilde{\pi}} = derive(M, \tilde{\pi})$
      $\hat{\pi} = determistic\_merge(M_{\tilde{\pi}})$
      if $consistent(M_{\hat{\pi}}, S^-)$
      then
        $M = M_{\hat{\pi}}$
        $\pi = \hat{\pi}$
        break
      end if
    end for
  end for
  return M
end

At each step i the algorithm attempts to refine the current partition by merging the blocks in order such that the quotient automaton corresponding to the refined partition is consistent with the negative sample S^-. The function $derive(M, \tilde{\pi})$

returns the quotient automaton $M_{\tilde{\pi}}$ corresponding to the partition $\tilde{\pi}$. Since $M_{\tilde{\pi}}$ might be a NFA, the function *determistic_merge*$(M_{\tilde{\pi}})$ returns the partition $\hat{\pi}$ obtained by successively merging the blocks in $\tilde{\pi}$ that cause non-determinism. The resulting FSA $(M_{\hat{\pi}})$ is guaranteed to be a DFA.

Let $||S^+||$ and $||S^-||$ denote the sums of the lengths of examples in S^+ and S^- respectively. $PTA(S^+)$ has $\mathcal{O}(||S^+||)$ states. The double loop of the algorithm performs $\mathcal{O}(||S^+||^2)$ state merges. Further, each time two blocks of the partition π are merged, the routine *deterministic_merge* in the worst case would cause $\mathcal{O}(||S^+||)$ state mergings and the function *consistent* that checks for the consistency of the derived DFA with the negative examples would incur a cost of $\mathcal{O}(||S^-||)$. Hence the time complexity of the RPNI algorithm is $\mathcal{O}((||S^+|| + ||S^-||) \cdot ||S^+||^2)$.

4 Learning from Simple Examples

The learnability of DFA under the standard PAC model is known to be hard. Restricting the underlying distribution of the PAC model to the universal distribution results in an interesting framework for PAC learning with *simple* examples. Concept classes such as $\log n$-term DNF and simple k-reversible DFA are PAC learnable with simple examples whereas their PAC learnability in the standard sense is unknown [Li & Vitányi, 1991]. Further, the learning system might be aided by a benign teacher who knows the target concept and uses this knowledge in selecting the examples. This framework due to [Denis *et al.*, 1996] is called the PACS model. Under this model, examples with low Kolmogorov complexity are called simple examples. Specifically, for a concept with representation r, the set $S^r_{sim} = \{\alpha \mid K(\alpha|r) \leq \mu lg(|r|)\}$ (where μ is a constant) is the set of simple examples for the concept. According to the universal distribution, simple examples have higher probability of being drawn. Formally, the probability of drawing an example α for a target concept with representation r is given as $\mathbf{m}_r(\alpha) = \lambda_r 2^{-K(\alpha|r)}$ where λ_r is a positive constant. Concept classes such as *poly-term* DNF and k-reversible DFA were shown to be learnable under the PACS model [Denis *et al.*, 1996]. We demonstrate the use of the RPNI algorithm under the PACS framework to provide a mechanism for efficiently learning DFA from simple examples.

A representative sample for a given concept is a set of examples that in some sense contains the necessary information for inference of the concept. For example, a characteristic sample for the target DFA A can be treated as its representative sample. The Occam's Razor theorem for the PACS model [Denis *et al.*, 1996] states that if there exists a representative sample of simple examples for each concept in a concept class then the concept class is PACS learnable. Let $\mathcal{C}$ be a concept class and $\mathcal{R}$ be the set of representations of $\mathcal{C}$. Consider a concept in $\mathcal{C}$ that has a representation r and the set S^r_{sim} of simple examples i.e. $\forall \alpha \in S^r_{sim}$, $K(\alpha|r) \leq \mu \lg(|r|)$. Further, let $S^r_{sim,rep}$ denote a set of simple representative examples for the concept r if such a set exists.

Lemma 1: (Due to [Denis *et al.*, 1996])
Suppose that a sample S is drawn according to $\mathbf{m}_r$. For an integer $l \geq |r|$, and $0 < \delta \leq 1$, if $|S| \geq \kappa l^{\mu}(\lg(1/\delta)+\lg(l^{\mu}))$, then with probability greater than $1-\delta$, $S^r_{sim} \subseteq S$ where κ is a constant.

Proof:
Claim 1.1: $\forall \alpha \in S^r_{sim},\ \mathbf{m}_r(\alpha) \geq \lambda_r l^{-\mu}$.

$$\begin{aligned}
\mathbf{m}_r(\alpha) &= \lambda_r 2^{-K(\alpha|r)} \\
&\geq \lambda_r 2^{-\mu \lg |r|} \\
&\geq \lambda_r |r|^{-\mu} \\
&\geq \lambda_r l^{-\mu}
\end{aligned}$$

Claim 1.2: $|S^r_{sim}| \leq 2l^{\mu}$.

$$\begin{aligned}
|S^r_{sim}| &\leq |\{\alpha \in \{0,1\}^* \mid K(\alpha|r) \leq \mu \lg(|r|)\}| \\
&\leq |\{\alpha \in \{0,1\}^* \mid K(\alpha|r) \leq \mu \lg(l)\}| \\
&\leq |\{\beta \in \{0,1\}^* \mid |\beta| \leq \mu \lg(l)\}| \\
&\leq 2^{\mu \lg(l)+1} \\
&\leq 2l^{\mu}
\end{aligned}$$

Claim 1.3: If $|S| \geq \kappa l^{\mu}(\lg(1/\delta) + \lg(l^{\mu}))$ then $\Pr(S^r_{sim} \subseteq S) \geq 1 - \delta$.

$$\begin{aligned}
\Pr(\alpha \in S^r_{sim} \text{ is not sampled in one random draw}) &\leq (1 - \lambda_r l^{-\mu}) \\
&\qquad \text{(claim 1.1)} \\
&\leq (1 - l^{-\mu}) \\
&\qquad \text{(since } \lambda_r \geq 1) \\
\Pr(\alpha \in S^r_{sim} \text{ is not sampled in } |S| \text{ random draws}) &\leq (1 - l^{-\mu})^{|S|} \\
\Pr(\text{some } \alpha \in S^r_{sim} \text{ is not sampled in } |S| \text{ random draws}) &\leq 2l^{\mu}(1 - l^{-\mu})^{|S|} \\
&\qquad \text{(claim 1.2)} \\
\Pr(S^r_{sim} \not\subseteq S) &\leq 2l^{\mu}(1 - l^{-\mu})^{|S|}
\end{aligned}$$

In order to get $\Pr(S^r_{sim} \subseteq S) \geq 1 - \delta$ we must have

$$\begin{aligned}
2l^{\mu}(1 - l^{-\mu})^{|S|} &\leq \delta \\
2l^{\mu}(e^{-l^{-\mu}})^{|S|} &\leq \delta, \quad \text{since } 1 - x \leq e^{-x} \\
\ln(2) + \ln(l^{\mu}) - |S| l^{-\mu} &\leq \ln(\delta) \\
|S| &\geq l^{\mu}\ (\ln(2) + \ln(l^{\mu}) + \ln(1/\delta)) \\
|S| &\geq \kappa l^{\mu}\ (\lg(l^{\mu}) + \lg(1/\delta)) \text{ where } \kappa \text{ is a constant}
\end{aligned}$$

Thus, $\Pr(S^r_{sim} \subseteq S) \geq 1 - \delta$ □

Corollary 1:
Suppose that a sample S is drawn according to $\mathbf{m}_r$. For an integer $l \geq |r|$, and

$0 < \delta \leq 1$, if $|S| \geq \kappa l^{\mu}(\lg(1/\delta)+\lg(l^{\mu}))$, then with probability greater than $1-\delta$, $S^r_{sim,rep} \subseteq S$ where κ is a constant.

Proof:
Follows immediately from Lemma 1 since $S^r_{sim,rep} \subseteq S^r_{sim}$. □

Next we demonstrate that for any DFA there exists a characteristic set of simple examples.

Lemma 2:
For any N state DFA with canonical encoding r ($|r| = \mathcal{O}(N\lg(N))$), there exists a characteristic sample of simple examples (denoted by $S^r_{sim,rep}$) such that each string of this sample is of length at most $2N-1$.

Proof:
Given the canonical encoding r of a DFA $A = (Q, \delta, \Sigma, q_0, F)$ it is possible to find a characteristic set of simple examples as follows:

1. Fix an enumeration of the shortest paths (in standard order) from the state q_0 to each state in Q except the dead state. This is the set of short prefixes of A. There are at most N such paths and each path is of length at most $N-1$.
2. Fix an enumeration of paths that includes each path identified above and its extension by each letter of the alphabet Σ. From the paths just enumerated retain only those that do not lead to a dead state of A. This represents the kernel of A. There are at most $N(|\Sigma|+1)$ such paths and each path is of length at most N.
3. The strings for $S^r_{sim,rep} = S^{r,+}_{sim,rep} \cup S^{r,-}_{sim,rep}$ are now evaluated as follows:
   (a) For each string α identified in step 2 above, determine the first suffix β in the standard enumeration of strings such that $\alpha\beta \in L(A)$. Since $|\alpha| \leq N$ and β is the shortest suffix in the standard order it is clear that $|\alpha\beta| \leq 2N-1$. Each such $\alpha\beta$ is a member of $S^{r,+}_{sim,rep}$.
   (b) For each pair of strings (α, β) in order where α is a string identified in step 1 and β is a string identified in step 2 determine the first suffix γ in the standard enumeration of strings such that $\alpha\gamma \in L(A)$ and $\beta\gamma \notin L(A)$ or vice versa. Since $|\alpha| \leq N-1$, $|\beta| \leq N$, and γ is the shortest distinguishing suffix for the states represented by α and β it is clear that $|\alpha\gamma|, |\beta\gamma| \leq 2N-1$. The accepted string from among $\alpha\gamma$ and $\beta\gamma$ is a member of $S^{r,+}_{sim,rep}$ and the rejected string is a member of $S^{r,-}_{sim,rep}$.

   It is clear that $|S^{r,+}_{sim,rep}| \leq (N^2+N)(|\Sigma|+1)$, $|S^{r,-}_{sim,rep}| \leq N^2(|\Sigma|+1)$, and the length of each string in S is less than $2N-1$.

Given a Turing machine TM (with knowledge of the target concept r) that implements the above algorithm for computing the set $S^r_{sim,rep}$, it is clear that given an index of length at most $\lg(3|\Sigma|N^2)$ bits TM will be able to extract the corresponding string belonging to $S^r_{sim,rep}$. Thus, $\forall \alpha \in S^r_{sim,rep}$

$$\begin{aligned} K(\alpha|r) &\leq \lg(3|\Sigma|N^2) \\ &\leq \mu \lg(|r|) \end{aligned}$$

This proves the lemma. □

Theorem 1:
For all N, the class $\mathcal{C}^{\leq N}$, of DFA whose canonical representations have at most N states is probably exactly learnable under the PACS model.

Proof:
Let A be a canonical DFA with at most N states and r be its canonical encoding. We define the simple representative sample $S^r_{sim,rep}$ to be the characteristic sample of A evaluated as described in lemma 2 above. Recall that the length of each example in $S^r_{sim,rep}$ is at most $2N-1$. Now consider the algorithm $\mathcal{A}$ that draws a sample S with the following properties:

1. $S = S^+ \cup S^-$ is a set of positive and negative examples corresponding to the target DFA A
2. The examples in S are drawn at random according to the distribution $\mathbf{m}_r$
3. $|S| \geq \kappa l^\mu (\lg(1/\delta) + \lg(l^\mu))$

Algorithm $\mathcal{A}$

Input: $N, 1/\delta$
Output: A DFA M

begin
  Randomly draw a labeled sample S according to $\mathbf{m}_r$
  Retain only those examples in S that have length at most $2N-1$
  $M = RPNI(S)$
  return(M)
end

Lemma 2 shows that there exists a characteristic set of simple examples ($S^r_{sim,rep}$) each of which is of length at most $2N-1$. From corollary 1 we know that with probability greater than $1-\delta$, $S^r_{sim,rep} \subseteq S$. From the correctness proof of the RPNI algorithm we know that if S is a superset of a characteristic sample for the canonical DFA A corresponding to the target then the DFA M returned by RPNI is equivalent to A. Since the size of S is polynomial in N and $1/\delta$ and the length of each string in S is restricted to $2N-1$, the RPNI algorithm and thus the algorithm $\mathcal{A}$ can be implemented to run in time polynomial in N and $1/\delta$.

Thus, the class $\mathcal{C}^{\leq N}$ is exactly learnable with probability greater than $1-\delta$ under the PACS model. □

We now show that the concept class of DFA can be learned in polynomial time under the PACS model. Let N be the number of states, and r be the encoding of the target A. Since N might not be known in advance we present a PAC learning algorithm $\mathcal{A}$ that iterates over successively larger guesses for N. At each step the algorithm draws a random sample according to $\mathbf{m}_r$, applies the RPNI algorithm to construct a DFA, and tests the DFA using another randomly drawn test sample. If the DFA is consistent with the test sample then the algorithm outputs the DFA and halts. Otherwise the algorithm continues with the next guess for N.

Theorem 2:
The concept class $\mathcal{C}$ of DFA is learnable in polynomial time under the PACS model.

Proof: We present the following learning algorithm for the PACS model and prove its correctness.

Algorithm $\mathcal{A}$

Input: ϵ, δ
Output: A DFA M

begin
1) $i = 1$, $EX = \phi$, $p(0, 1/\delta) = 0$
2) **Repeat**
    Draw $p(i, 1/\delta) - p(i-1, 1/\delta)$ examples according to $\mathbf{m}_r$
    Add the examples just drawn to the set EX
    Let S be the subset of examples in EX of length at most $2i - 1$
    $M = RPNI(S)$
    Draw $q(i, 1/\epsilon, 1/\delta)$ examples according to $\mathbf{m}_r$ and call this set T
    If $consistent(M, T)$
    Then Output M and **halt**
    Else $i = i + 1$
  until eternity
end

In the above algorithm the polynomial p is defined such that a sample S of size $p(N, 1/\delta)$ contains a set of simple representative examples $S^r_{sim,rep}$ with probability greater than $1 - \delta$. Note that according to corollary 1, $p(N, 1/\delta) = \mathcal{O}(l^{\mu}\ (\lg(l^{\mu}) + \lg(1/\delta)))$ will satisfy this constraint. The polynomial q is defined as $q(i, 1/\epsilon, 1/\delta) = \frac{1}{\epsilon}[2\ln(i+1) + \ln(\frac{1}{\delta})]$.

From lemma 2, it is clear that for any step i of the algorithm where $i \geq N$, the set S will include a characteristic sample of simple examples ($S^r_{sim,rep}$) with probability greater than $1 - \delta$. In this case the RPNI algorithm will return a DFA M that is equivalent to the target A and hence M will be consistent with the test sample T. Thus, the algorithm will halt and correctly output the target

DFA with probability greater than $1-\delta$.

Consider the probability that the algorithm halts at some step i and returns a DFA M with error greater than ϵ.

$$
\begin{aligned}
\Pr(M \text{ and } A \text{ are consistent on some } \alpha) &\leq 1-\epsilon \\
\Pr(M \text{ and } A \text{ are consistent on all } \alpha \in T) &\leq (1-\epsilon)^{|T|} \\
&\leq (1-\epsilon)^{\frac{1}{\epsilon}[2\ln(i+1)+\ln(\frac{1}{\delta})]} \\
&\leq e^{-[2\ln(i+1)+\ln(\frac{1}{\delta})]} \text{ since } 1-x \leq e^{-x} \\
&\leq \frac{\delta}{(i+1)^2}
\end{aligned}
$$

The probability that the algorithm halts at any step i and returns a DFA with error greater than ϵ is at most $\sum_{i=1}^{\infty} \frac{\delta}{(i+1)^2}$ which can be shown to be strictly less than δ. Thus, we have shown that with probability greater than $1-\delta$ the algorithm returns a DFA with error at most ϵ. Further, it is easy to see that the running time of the algorithm is polynomial in N, $|\Sigma|$, $1/\epsilon$, $1/\delta$, and m (where m is the length of the longest test example seen by the algorithm). Thus, the class of DFA is efficiently PAC learnable under the PACS model. □

5 Discussion

The problem of exactly learning the target DFA from an arbitrary set of labeled examples and the problem of approximating the target DFA from labeled examples under Valiant's PAC learning framework are both known to be hard problems. Thus, the question as to whether DFA are efficiently learnable under some restricted yet fairly general and practically useful classes of distributions was clearly of interest. In this paper, we have answered an important variant of this question in the affirmative by providing a framework for efficient PAC learning of DFA from simple examples.

In particular, we have shown that if a benign teacher (or nature) provides labeled examples of a target DFA drawn according to the universal distribution $\mathbf{m}_r$, then with probability at least $1-\delta$, this sample is guaranteed to include a characteristic set of examples for the target DFA when the sample size is polynomial in the number of states of the target DFA and $\lg(\frac{1}{\delta})$. The RPNI algorithm is guaranteed to return a DFA that is equivalent to the target DFA provided the set of examples given to the algorithm includes a characteristic set. Thus, by casting the RPNI algorithm in the framework of learning from simple examples we demonstrate probably exact leanability of DFA from labeled examples. Further, we show that when an upper bound on the number of states of the target DFA is unknown, we can use this algorithm iteratively, to efficiently PAC learn the concept class of DFAs with any desired error and confidence parameters.

The class of simple distributions includes a large variety of probability distributions (including all computable distributions that are characterized by finite precision parameters). Li and Vitanyí showed that a concept class is efficiently learnable under the universal distribution if and only if it is efficiently learnable under each simple distribution provided the sampling is done according to the universal distribution [Li & Vitányi, 1991]. This raises the possibility of using sampling under the universal distribution to learn under all computable distributions. However, the universal distribution is not computable. Whether one can instead get by with a polynomially computable approximation of the universal distribution remains an open question. It is known that the universal distribution for the class of polynomially-time bounded simple distributions is computable in exponential time [Li & Vitányi, 1991]. This raises a number of interesting possibilities for learning under simple distributions.

A related question of interest has to do with the nature of environments that can be modeled by simple distributions. Learning scenarios that involve biological intelligent systems (e.g., language acquisition) appear to benefit from simple examples. It is not inconceivable that these systems first process simpler examples and then as they mature, take on the more complex examples. In particular, if Kolmogorov complexity is an appropriate measure of the intrinsic complexity of objects in nature and if nature (or the teacher) has a propensity for simplicity, then it stands to reason that the examples presented to the learner by the environment are likely to be generated by a simple distribution. Against this background, empirical evaluation of the performance of the proposed algorithms using examples that come from natural domains is clearly of interest.

In the incremental version of the RPNI algorithm [Dupont, 1996a] the learner maintains a hypothesis that is consistent with all labeled examples seen thus far and modifies it whenever a new inconsistent example is observed. The convergence of this algorithm relies on the fact that sooner or later, the set of labeled examples seen by the learner will include a characteristic set. If in fact the stream of examples provided to the learner is drawn according to a simple distribution, our results show that in an incremental setting the characteristic set would be made available relatively early (during learning) with a sufficiently high probability and hence the algorithm will converge quickly to the desired target.

Some of the negative results in approximate identification of DFA available in the literature were derived by showing that an efficient algorithm for learning DFA would entail algorithms for solving known hard problems such as *learning boolean formulae* [Pitt & Warmuth, 1988] and *breaking the RSA cryptosystem* [Kearns & Valiant, 1989]. It would be interesting to explore the implications (if any) of our results on efficient learning of DFA from simple examples on these problems.

Acknowledgements

The authors wish to thank Dr. Jack Lutz for introducing them to Kolmogorov complexity, Dr. Giora Slutzki for several helpful discussions on automata theory, and Dr. Pierre Dupont for a careful review of an earlier draft of this paper and several helpful suggestions.

References

Angluin, D. (1981). A Note on the Number of Queries Needed to Identify Regular Languages. *Information and Control*, **51**, 76–87.

Angluin, D. (1987). Learning Regular Sets from Queries and Counterexamples. *Information and Computation*, **75**, 87–106.

Chomsky, N. (1956). Three Models for the Description of Language. *PGIT*, **2**(3), 113–124.

Denis, F., D'Halluin, C., & Gilleron, R. (1996). PAC Learning with Simple Examples. *STACS'96 - Proceedings of the* 13^{th} *Annual Symposium on the Theoretical Aspects of Computer Science*, 231–242.

Dupont, P. (1996a). Incremental Regular Inference. *Pages 222–237 of:* Miclet, L., & Higuera, C. (eds), *Proceedings of the Third ICGI-96, Lecture Notes in Artificial Intelligence 1147*. Montpellier, France: Springer.

Dupont, P. (1996b). *Utilisation et Apprentissage de Modèles de Language pour la Reconnaissance de la Parole Continue.* Ph.D. thesis, Ecole Normale Supérieure des Télécommunications, Paris, France.

Dupont, P., Miclet, L., & Vidal, E. (September 1994). What is the Search Space of the Regular Inference? *Pages 25–37 of: Proceedings of the Second International Colloquium on Grammatical Inference (ICGI'94).*

Gold, E. M. (1978). Complexity of Automaton Identification from Given Data. *Information and Control*, **37**(3), 302–320.

Hopcroft, J., & Ullman, J. (1979). *Introduction to Automata Theory, Languages, and Computation.* Addison Wesley.

Kearns, M., & Valiant, L. G. (May 1989). Cryptographic Limitations on Learning Boolean Formulae and Finite Automata. *Pages 433–444 of: Proceedings of the* 21^{st} *Annual ACM Symposium on Theory of Computing.* ACM, New York.

Lang, K. J. (1992). Random DFA's can be approximately learned from sparse uniform sample. *Pages 45–52 of: Proceedings of the 5th ACM workshop on Computational Learning Theory.*

Li, M., & Vitányi, P. (1991). Learning Simple Concepts under Simple Distributions. *SIAM Journal of Computing*, **20**, 911–935.

Li, M., & Vitányi, P. (1997). *An Introduction to Kolmogorov Complexity and its Applications,* 2^{nd} *edition.* New York: Springer Verlag.

Oncina, J., & García, P. (1992). Inferring Regular Languages in Polynomial Update Time. *Pages 49–61 of:* Pérez, N. *et al* (ed), *Pattern Recognition and Image Analysis.* World Scientific.

Pao, T., & Carr, J. (1978). A Solution of the Syntactic Induction-Inference Problem for Regular Languages. *Computer Languages*, **3**, 53–64.

Parekh, R.G., & Honavar, V. G. (August 1993). Efficient Learning of Regular Languages using Teacher Supplied Positive Examples and Learner Generated Queries. *Pages 195–203 of: Proceedings of the Fifth UNB Conference on AI.*

Pitt, L. (1989). Inductive Inference, DFAs and Computational Complexity. *Pages 18–44 of: Analogical and Inductive Inference, Lecture Notes in Artificial Intelligence 397.* Springer-Verlag.

Pitt, L., & Warmuth, M. K. (1988). Reductions among prediction problems: on the difficulty of predicting automata. *Pages 60–69 of: Proceedings of the* 3^{rd} *I.E.E.E. Conference on Structure in Complexity Theory.*

Pitt, L., & Warmuth, M. K. (1989). The minimum consistency DFA problem cannot be approximated within any polynomial. *Pages 421–432 of: Proceedings of the* 21^{st} *ACM Symposium on the Theory of Computing.* ACM.

Trakhtenbrot, B., & Barzdin, Ya. (1973). *Finite Automata: Behavior and Synthesis.* Amsterdam: North Holland Publishing Company.

Valiant, L. (1984). A Theory of the Learnable. *Communications of the ACM*, **27**, 1134–1142.