Theoretical Guarantees for Learning Weighted Automata

Borja Balle

ICGI Keynote — October 2016
Thanks To My Collaborators!

Mehryar Mohri
Ariadna Quattoni
Xavier Carreras
Prakash Panangaden
Doina Precup
Outline

1. The Complexity of PAC Learning Regular Languages

2. Empirical Risk Minimization for Regular Languages

3. Statistical Learning of Weighted Automata via Hankel Matrices
Outline

1. The Complexity of PAC Learning Regular Languages

2. Empirical Risk Minimization for Regular Languages

3. Statistical Learning of Weighted Automata via Hankel Matrices
Regular Inference (Informal Description)

- Unknown regular language $L \subseteq \Sigma^*$
  - With indicator function $f : \Sigma^* \to \{0, 1\}$
- Given examples $(x^1, f(x^1)), (x^2, f(x^2)), \ldots$
  - Finite or infinite
  - (positive and negative) OR (only positive)
- Find a representation for $L$ (eg. a DFA)
  - Using a reasonable amount of computation
  - After seeing a reasonable amount of examples
PAC Learning Regular Languages

- Concept class $\mathcal{C}$ of functions $\Sigma^* \rightarrow \{0, 1\}$
  - Eg. $\mathcal{C} = \text{DFA}_n$ all regular languages recognized by DFA with $n$ states
- Hypothesis class $\mathcal{H}$ of representations for functions $\Sigma^* \rightarrow \{0, 1\}$
  - Proper learning $\mathcal{H} = \mathcal{C}$
  - Improper learning $\mathcal{H} \neq \mathcal{C}$

Definition: PAC Learner

An algorithm $A$ such that for any $f \in \mathcal{C}$ and any prob. dist. $D$ on $\Sigma^*$, and any accuracy $\varepsilon$ and confidence $\delta$, satisfies: given a large enough sample of examples $S = ((x^i, f(x^i)))$ i.i.d. from $D$, the output hypothesis $\hat{f} = A(S) \in \mathcal{H}$ satisfies $P_{x \sim D}[f(x) \neq \hat{f}(x)] \leq \varepsilon$ with probability at least $1 - \delta$.

- Large enough typically means polynomial of $1/\varepsilon$, $1/\delta$, size of $f$
- For any prob. dist. $D$ on $\Sigma^*$ is called distribution-free learning

Note: see [De la Higuera, 2010] for other important formal learning models
# Sample Complexity of PAC Learning DFA

## Sample Complexity

The distribution-free sample complexity of PAC learning $C = \mathcal{DFA}_n$ is polynomial in $n$ and $|\Sigma|$.

Follows from:

- Any concept class $C$ can be proper PAC-learned with:
  - $|S| = O\left(\frac{\text{VC}(C) \log(1/\varepsilon) + \log(1/\delta)}{\varepsilon}\right)$ [Vapnik, 1982]
  - $|S| = O\left(\frac{\text{VC}(C) \log(1/\delta)}{\varepsilon}\right)$ [Haussler et al., 1994]
  - $|S| = O\left(\frac{\text{VC}(C) + \log(1/\delta)}{\varepsilon}\right)$ [Hanneke, 2016]

- $\text{VC}(\mathcal{DFA}_n) = O(|\Sigma|n \log n)$ [Ishigami and Tani, 1993]

## Generic Learning Algorithm:

- Upper bounds in [Vapnik, 1982, Hanneke, 2016] apply to consistent learning algorithms

- $A$ is consistent if for any sample $S = ((x^i, f(x^i)))$ the hypothesis $\hat{f} = A(S)$ satisfies $\hat{f}(x^i) = f(x^i)$ for all $i$
Computational Complexity of PAC Learning DFA

- Proper PAC learning of DFA is equivalent to finding *smallest consistent DFA* with $S$ [Board and Pitt, 1992]
- Approximating the smallest consistent DFA is NP-hard [Pitt and Warmuth, 1993, Chalermsook et al., 2014]
- Improper learning DFA is as hard as breaking RSA [Kearns and Valiant, 1994]
- Improper learning DFA is as hard as refuting random CSP [Daniely et al., 2014]
Is Worst-case Hardness Too Pessimistic?

Positive Results:

- Given characteristic sample, state-merging can find smallest consistent DFA [Oncina and García, 1992]
- PAC learning is possible under nice distributions adapted to target language [Parekh and Honavar, 2001, Clark and Thollard, 2004]
- Random DFA under uniform distributions seem easy to learn [Lang, 1992, Angluin and Chen, 2015]
- And also lots of successful heuristics in practice: EDSM, SAT solvers, etc.

Take Away:

- By giving up on distribution-free and focusing on *nice distributions* efficient PAC learning is possible
- Almost all of these algorithms still focus on *sample consistency*
- Do we expect them to work well for *practical applications*?
  - Probably yes for software engineering
  - Probably not for NLP, robotics, bioinformatics, ...
Outline

1. The Complexity of PAC Learning Regular Languages

2. Empirical Risk Minimization for Regular Languages

3. Statistical Learning of Weighted Automata via Hankel Matrices
Regular Inference as an Optimization Problem

Thought Experiment

Given input sample $S = ((x^i, y^i))$ for $i = 1 \ldots 100$, would you rather:

1. classify all 100 examples correctly with 50 states, or
2. classify 95 examples correctly with 5 states?

Optimization Problems

1. Minimal consistent DFA

$$
\min_{A \in \mathcal{DFA}} |A| \quad \text{s.t.} \quad A(x^i) = y^i \ \forall i \in [m]
$$

2. Empirical risk minimization in $\mathcal{DFA}_n$

$$
\min_{A \in \mathcal{DFA}} \frac{1}{m} \sum_{i=1}^{m} 1[A(x^i) \neq y^i] \quad \text{s.t.} \quad |A| \leq n
$$
**Statistical Learning for Classification**

**Statistical Learning Setup**

- $\mathcal{D}$ probability distribution over $\Sigma^* \times \{+1, -1\}$
- $\mathcal{H}$ hypothesis class of functions $\Sigma^* \rightarrow \{+1, -1\}$
- $\ell_{01}$ the 0-1 loss function for $y, \hat{y} \in \{+1, -1\}$

\[
\ell_{01}(\hat{y}, y) = \frac{1 - \text{sign}(\hat{y}y)}{2} = 1[\hat{y} = y]
\]

**Statistical Learning Goal**

- Find the minimizer of the *average loss*:

\[
f^* = \arg\min_{f \in \mathcal{H}} \mathbb{E}_{(x,y) \sim \mathcal{D}} [\ell_{01}(f(x), y)] = \arg\min_{f \in \mathcal{H}} \mathbb{L}_\mathcal{D}(f; \ell_{01})
\]

- From a sample $S = ((x^i, y^i))$ with $m$ i.i.d. examples from $\mathcal{D}$

\[
\mathbb{E}_{(x,y) \sim \mathcal{D}} [\ell_{01}(f(x), y)] \approx \frac{1}{m} \sum_{i=1}^{m} \ell_{01}(f(x^i), y^i)
\]
ERM and VC Theory

Empirical Risk Minimization (ERM)

- Given the sample $S = ((x^i, y^i))$ return the hypothesis

$$
\hat{f} = \operatorname{argmin}_{f \in \mathcal{H}} \frac{1}{m} \sum_{i=1}^{m} \ell_{01}(f(x^i), y^i, ) = \operatorname{argmin}_{f \in \mathcal{H}} \hat{L}_S(f; \ell_{01})
$$

Statistical Justification

- Generalization bound based on VC theory: with prob. at least $1 - \delta$ over $S$ (e.g. see [Mohri et al., 2012])

$$
L_D(f; \ell_{01}) \leq \hat{L}_S(f; \ell_{01}) + O \left( \sqrt{\frac{\text{VC}(\mathcal{H}) \log m + \log(1/\delta)}{m}} \right) \quad \forall f \in \mathcal{H}
$$

- In the case $\mathcal{H} = \text{DFA}_n$:

$$
L_D(A; \ell_{01}) \leq \hat{L}_S(A; \ell_{01}) + O \left( \sqrt{\frac{\sum |n \log n \log m + \log(1/\delta)}{m}} \right) \quad \forall |A| \leq n
$$
Sources of Hardness in ERM for DFA

\[ \min_{A \in \mathcal{DFA}} \frac{1}{m} \sum_{i=1}^{m} \ell_{01}(A(x^i), y^i) \quad \text{s.t.} \quad |A| \leq n \]

- Non-convex loss: \( \ell_{01}(A(x), y) \) is not convex in \( A(x) \) because of sign
- Combinatorial search space: search over DFA is search over labelled directed graph with constraints
- Non-convex constraint: introducing \( |A| \) into the optimization is hard

**Common Wisdom:** Optimization tools that work better in practice deal with differentiable and/or convex problems
Roadmap to a Tractable Surrogate

- Replace by $\ell_{01}$ by a convex upper bound

- Make search space continuous: from DFA to WFA

- Identify convex constraints on WFA that can prevent overfitting
Writing DFA with Matrices and Vectors

\[ A = \langle \alpha, \beta, \{A_\sigma\} \rangle \]

\[ \alpha = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \beta = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad A_a = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix} \quad A_b = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix} \]

\[ A(aab) = \alpha^\top A_a A_a A_b \beta = 1 \]
Weighted Finite Automata (WFA)

\[ A = \langle \alpha, \beta, \{ A_\sigma \} \rangle \]

\[ \alpha = \begin{bmatrix} -1 \\ 0.5 \end{bmatrix} \quad \beta = \begin{bmatrix} 1.2 \\ 0 \end{bmatrix} \quad A_a = \begin{bmatrix} 1.2 & -1 \\ -2 & 3.2 \end{bmatrix} \quad A_b = \begin{bmatrix} 2 & -2 \\ 0 & 5 \end{bmatrix} \]

\[ A : \Sigma^* \to \mathbb{R} \quad A(x_1 \cdots x_T) = \alpha^\top A_{x_1} \cdots A_{x_T} \beta \]
**ERM for WFA is Differentiable**

$$\min_{A \in \mathcal{WFA}} \frac{1}{m} \sum_{i=1}^{m} \ell(A(x^i), y^i) \quad \text{s.t.} \quad |A| = n$$

with loss $\ell(\hat{y}, y)$ **differentiable on first coordinate**

### Gradient Computation

- **WFA** $A = \langle \alpha, \beta, \{A_\sigma\} \rangle$, $x \in \Sigma^*$, $y \in \mathbb{R}$, can compute $\nabla_A \ell(A(x), y)$
- Example with $x = abc a$ and weights in $A_a$:

$$\nabla_{A_a} \ell(A(x), y) = \frac{\partial \ell}{\partial \hat{y}}(A(x), y) \cdot (\nabla_{A_a} \alpha^T A_a A_b A_c A_a \beta)$$

$$= \frac{\partial \ell}{\partial \hat{y}}(A(x), y) \cdot \left( \alpha \beta^T A_a^T A_c A_b + A_c A_b A_a \alpha \beta^T \right)$$

- Can use gradient descent to “solve” ERM for WFA
- The optimization is highly non-convex, but its commonly done in RNN
- Since $\mathcal{WFA}_n$ is infinite, what is a proper way to prevent overfitting?
The risk of overfitting can be controlled with generalization bounds of the form: for any $D$, with prob. $1 - \delta$ over $S \sim D^m$

$$L_D(f; \ell) \leq \hat{L}_S(f; \ell) + C(S, H, \ell) \quad \forall f \in H$$

Rademacher complexity provides bounds for any $H = \{f : \Sigma^* \to \mathbb{R}\}$

$$R_m(H) = E_{S \sim D^m} E_\sigma \left[ \sup_{f \in H} \frac{1}{m} \sum_{i=1}^m \sigma_i f(x^i) \right] \text{ where } \sigma_i \sim \text{unif}(\{+1, -1\})$$

For a bounded Lipschitz loss $\ell$ with probability $1 - \delta$ over $S \sim D^m$ (e.g. see [Mohri et al., 2012])

$$L_D(f; \ell) \leq \hat{L}_S(f; \ell) + O \left( R_m(H) + \sqrt{\frac{\log(1/\delta)}{m}} \right) \quad \forall f \in H$$
Rademacher Complexity of WFA

- Given a pair of Hölder conjugate integers $p, q$ ($1/p + 1/q = 1$), define a norm on WFA given by

$$\|A\|_{p,q} = \max \left\{ \|\alpha\|_p, \|\beta\|_q, \max_{a \in \Sigma} \|A_a\|_q \right\}$$

- Let $A_n \subset WFA_n$ be the class of WFA with $n$ states given by

$$A_n = \{ A \in WFA_n \mid \|A\|_{p,q} \leq 1 \}$$

**Theorem [Balle and Mohri, 2015b]**

The Rademacher complexity of $A_n$ is bounded by

$$\mathcal{R}_m(A_n) = O \left( \frac{L_m}{m} + \sqrt{\frac{n^2|\Sigma|\log(m)}{m}} \right),$$

where $L_m = \mathbb{E}_S[\max_i |x^i|]$. 
Learning WFA with Gradient Descent

- Solve the following ERM problem with (stochastic) projected gradient descent:

\[
\min_{A \in \mathcal{W} \mathcal{F} \mathcal{A}_n} \frac{1}{m} \sum_{i=1}^{m} \ell(A(x^i), y^i) \quad \text{s.t.} \quad \|A\|_{p, q} \leq R
\]

- Control overfitting by tuning $R$ (e.g. via cross-validation)

- Can equally solve classification ($y^i \in \{+1, -1\}$) and regression ($y^i \in \mathbb{R}$) with differentiable loss functions

- Risk of underfitting: unlikely that we will find the global optimum, might get stuck in local optimum
Outline

1. The Complexity of PAC Learning Regular Languages

2. Empirical Risk Minimization for Regular Languages

3. Statistical Learning of Weighted Automata via Hankel Matrices
Hankel Matrices and Fliess’ Theorem

Given \( f: \Sigma^* \rightarrow \mathbb{R} \) define its Hankel matrix \( H_f \in \mathbb{R}^{\Sigma^* \times \Sigma^*} \) as

\[
\begin{bmatrix}
\varepsilon & a & b & \ldots & s & \ldots \\
\varepsilon & f(\varepsilon) & f(a) & f(b) & \vdots \\
a & f(a) & f(aa) & f(ab) & \vdots \\
b & f(b) & f(ba) & f(bb) & \vdots \\
\vdots & & & & \ddots \\
p & \ldots & \ldots & \ldots & \ldots & f(ps) \\
\vdots & & & & & \ddots
\end{bmatrix}
\]

**Theorem [Fliess, 1974]**

The rank of \( H_f \) is finite if and only if \( f \) is computed by a WFA, in which case \( \text{rank}(H_f) \) equals the number of states of a minimal WFA computing \( f \)
From Hankel to WFA

\[ A(p_1 \cdots p_T s_1 \cdots s_{T'}) = \alpha^T A p_1 \cdots A p_T A s_1 \cdots A s_{T'}, \beta \]

\[ p \begin{bmatrix} \vdots & f(p s) & \vdots \end{bmatrix} = \begin{bmatrix} \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} \vdots & \vdots & \vdots \end{bmatrix} \]

\[ A(p_1 \cdots p_T a s_1 \cdots s_{T'}) = \alpha^T A p_1 \cdots A p_T A_a A s_1 \cdots A s_{T'}, \beta \]

\[ p \begin{bmatrix} \vdots & f(p a s) & \vdots \end{bmatrix} = \begin{bmatrix} \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} \vdots & \vdots & \vdots \end{bmatrix} \]

- Algebraically: \( H = PS \) and \( H_a = PA_a S \), so we can learn by \( A_a = P^+H_a S^+ \)
- This is the underlying principle behind query learning and spectral learning for WFA [Balle and Mohri, 2015a]
- For more information, see our EMNLP’14 tutorial with A. Quattoni and X. Carreras [Balle et al., 2014]
**Learning with Hankel Matrices** [Balle and Mohri, 2012]

**Step 1:** Learn a finite Hankel matrix over $\mathcal{P} \times \mathcal{S}$ directly from data by solving the convex ERM

$$
\hat{H} = \arg\min_{H \in \mathbb{R}^{\mathcal{P} \times \mathcal{S}}} \frac{1}{m} \sum_{i=1}^{m} \ell(H(x^i), y^i) \quad \text{s.t. } H \in \text{Hankel}
$$

**Step 2:** Extract sub-blocks $\hat{H}_e, \hat{H}_a$ from the Hankel matrix $\hat{H}$

$$
\mathcal{P} \subseteq \mathcal{P}_e \cup (\mathcal{P}_e \cdot \Sigma)
$$

$$
\hat{H}_e(p, s) = \hat{H}(p, s) \quad p \in \mathcal{P}_e, s \in \mathcal{S}
$$

$$
\hat{H}_a(p, s) = \hat{H}(pa, s) \quad p \in \mathcal{P}_e, s \in \mathcal{S}
$$

**Step 3:** Learn a WFA from the Hankel matrix using SVD

$$
\hat{H}_e = UDV^\top
$$

$$
\hat{A}_a = U^\top \hat{H}_a VD^{-1}
$$
Controlling Overfitting with Hankel Matrices

- To prevent overfitting, control number of states of resulting WFA by

\[
\hat{H} = \arg\min_{H \in \mathbb{R}^{p \times s}} \frac{1}{m} \sum_{i=1}^{m} \ell(H(x^i), y^i) \quad \text{s.t. } H \in \text{Hankel}, \quad \text{rank}(H) \leq n
\]

- Since this is not convex, a usual surrogate is to use Schatten norms

\[
\hat{H} = \arg\min_{H \in \mathbb{R}^{p \times s}} \frac{1}{m} \sum_{i=1}^{m} \ell(H(x^i), y^i) \quad \text{s.t. } H \in \text{Hankel}, \quad \|H\|_{S,p} \leq R
\]

where \( \|H\|_{S,p} = \|(s_1, \ldots, s_n)\|_p \) and \( s_1 \geq \cdots \geq s_n > 0 \) are the singular values of \( H \)

- These norms can be computed in polynomial time even for infinite Hankel matrices [Balle et al., 2015]
Rademacher Complexity of Hankel Matrices

Given $R > 0$ and $p \geq 1$ define the class of infinite Hankel matrices

$$\mathcal{H}_p = \left\{ H \in \mathbb{R}^{\Sigma^* \times \Sigma^*} \mid H \in \text{Hankel}, \|H\|_{S,p} \leq R \right\}$$

**Theorem [Balle and Mohri, 2015b]**

The Rademacher complexity of $\mathcal{H}_2$ is bounded by

$$\mathcal{R}_m(\mathcal{H}_2) = O \left( \frac{R}{\sqrt{m}} \right).$$

The Rademacher complexity of $\mathcal{H}_1$ is bounded by

$$\mathcal{R}_m(\mathcal{H}_1) = O \left( \frac{R \log(m) \sqrt{W_m}}{m} \right),$$

where $W_m = E_S \left[ \min_{\text{split}(S)} \max \{ \max_p \sum_i 1[p^i = p], \max_s \sum_i 1[s^i = s] \} \right]$.

Note: split$(S)$ contains all possible prefix-suffix splits $x^i = p^i s^i$ of all strings in $S$.
Constrained vs. Regularized Optimization

- Constrained ERM with parameter $R > 0$
  \[
  \min_{H \in \mathbb{R}^{p \times S}} \frac{1}{m} \sum_{i=1}^{m} \ell(H(x^i), y^i) \quad \text{s.t. } H \in \text{Hankel}, \quad \|H\|_{S,p} \leq R
  \]

- Regularized ERM with parameter $\lambda > 0$
  \[
  \min_{H \in \mathbb{R}^{p \times S}} \frac{1}{m} \sum_{i=1}^{m} \ell(H(x^i), y^i) + \lambda \|H\|_{S,p} \quad \text{s.t. } H \in \text{Hankel}
  \]

- Regularized versions can be easier to solve and $\lambda$ easier to tune
- For example, for $\mathcal{H}_2$ bounds *informally* say that for any $H$
  \[
  L_D(H; \ell) \leq \hat{L}_S(H; \ell) + O\left(\frac{\|H\|_{S,2}}{\sqrt{m}}\right)
  \]
  so choosing $\lambda = O(1/\sqrt{m})$ would imply ERM minimizes a direct upper bound on $L_D$
Applications of Learning with Hankel Matrices

- Max-margin taggers [Quattoni et al., 2014]

- Unsupervised transducers [Bailly et al., 2013b]

- Unsupervised WCFG [Bailly et al., 2013a]
It is possible to solve regular inference with machine learning, focusing on the realistic statistical learning scenario, and still obtain meaningful theoretical guarantees.

In practice works very well, but convex algorithms are not always scalable: we need good implementations.

How to choose $P$ and $S$ from data in practice?

PAC learning of WFA for regression is still open.

Theoretical link between finite and infinite Hankel matrices is still weak.


In *EMNLP*.

In *NIPS*.

In *NIPS*.
Learning weighted automata.

On the rademacher complexity of weighted automata.

A canonical form for weighted automata and applications to approximate minimization.
In *LICS*.

Spectral Learning Techniques for Weighted Automata, Transducers, and Grammars.
References III

On the necessity of occam algorithms.

Pre-reduction graph products: Hardnesses of properly learning dfas and approximating edp on dags.

Partially distribution-free learning of regular languages from positive samples.

From average case complexity to improper learning complexity.
*Grammatical inference: learning automata and grammars.*
Cambridge University Press.

Matrices de Hankel.
*Journal de Mathématiques Pures et Appliquées.*

Complexity of automaton identification from given data.

The optimal sample complexity of pac learning.

Predicting \{0, 1\}-functions on randomly drawn points.
References V

The vc-dimensions of finite automata with n states.

Cryptographic limitations on learning boolean formulae and finite automata.

Lang, K. J. (1992).
Random dfa’s can be approximately learned from sparse uniform examples.
In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 45–52. ACM.

*Foundations of machine learning*.
MIT press.

Identifying regular languages in polynomial time.
References VI

Learning dfa from simple examples.

The minimum consistent dfa problem cannot be approximated within any polynomial.

Spectral regularization for max-margin sequence tagging.
In *ICML*.

Estimation of dependencies based on empirical data.
Theoretical Guarantees for Learning Weighted Automata

Borja Balle

Mathematics & Statistics | Lancaster University

ICGI Keynote — October 2016