PAC learning of Probabilistic Automaton based on the Method of Moments

Hadrien Glaude, Olivier Pietquin

PAC learning of Probabilistic Automaton based on the Method of Moments

Hadrien Glaude HADRIEN.GLAUDE@INRIA.FR
Univ. Lille, CRIStAL, UMR 9189, SequeL Team, Villeneuve d'Ascq, 59650, France

Olivier Pietquin[1] OLIVIER.PIETQUIN@UNIV-LILLE1.FR
Institut Universitaire de France (IUF), Univ. Lille, CRIStAL, UMR 9189, SequeL Team, Villeneuve d'Ascq, 59650, France

Abstract

Probabilitic Finite Automata (PFA) are generative graphical models that define distributions with latent variables over finite sequences of symbols, a.k.a. stochastic languages. Traditionally, unsupervised learning of PFA is performed through algorithms that iteratively improves the likelihood like the Expectation-Maximization (EM) algorithm. Recently, learning algorithms based on the so-called Method of Moments (MoM) have been proposed as a much faster alternative that comes with PAC-style guarantees. However, these algorithms do not ensure the learnt automata to model a proper distribution, limiting their applicability and preventing them to serve as an initialization to iterative algorithms. In this paper, we propose a new MoM-based algorithm with PAC-style guarantees that learns automata defining proper distributions. We assess its performances on synthetic problems from the PAutomaC challenge and real datasets extracted from Wikipedia against previous MoM-based algorithms and EM algorithm.

1. Introduction

In this paper, we address the problem of learning a distribution with latent variables over sequences of symbols from independent samples drawn from it. In particular, we are interested in distributions realized by generative graphical models called Probabilistic Finite Automata (PFA). Traditionally, algorithms to learn PFA rely on iterative procedures that maximize the joint likelihood like the gradient ascent or EM and its variants. However, these algorithms do not scale well with the number of samples and latent variables to the point where obtaining good solutions for large models becomes intractable. In addition, they are prone to get stuck in local optima. There exist also full Bayesian methods like variational Bayes or Gibbs sampling, but these methods are computationally expensive and strongly rely on assumptions made on the prior.

A recent alternative line of work consists in modeling the distribution at sight by a Multiplicity Automaton (MA), also called weighted finite automaton (Balle, 2013). MA are graphical models that realize functions over finite sequences of symbols. Hence, they encompass a large variety of linear sequential systems in addition to stochastic languages (Thon & Jaeger, 2015). For example, when MA model stochastic processes, they are equivalent to Observable Operator Models (Thon & Jaeger, 2015). When considering action-observation pairs as symbols, MA can model controlled processes and are equivalent to Predictive State Representation (Glaude et al., 2014). In fact, MA are strictly more general and infinitely more compact than PFA, Hidden Markov Models (HMMs) and Partially Observable Markov Decision Processes (POMDPs). Casting the learning problem into the one of learning a more general class of models allows using the MoM. This method leverages the fact that low order moments of distributions contain most of the distribution information and are typically easy to estimate. MoM-based algorithms have several pros over iterative methods. First, they are extremely fast. Their complexity is linear in the number of samples as estimated moments can be computed in one pass on the training set. The time complexity is also polynomial in the learnt model size as these algorithms rely only on few linear algebra operations to recover the parameters. In addition, MoM-based algorithms are often consistent with theoretical guarantees in the form of finite-sample bounds on the ℓ_1 error between the learnt function and the target distribution. Thus, these algorithms are Probably Approximately Correct (PAC) in a sense defined by (Kearns et al., 1994), if we allow the number of samples to depend polynomially on some parameters measuring the complexity of the target

Proceedings of the 33^{rd} International Conference on Machine Learning, New York, NY, USA, 2016. JMLR: W&CP volume 48.

[1] now with Google DeepMind

distribution. These parameters are typically small singular values of matrices defined by the target distribution.

However, current MoM-based algorithms have a major con. Although errors in the estimated parameters are bounded, they may correspond to automatons that lie outside the class of models defining proper distributions. Hence, the learnt models can output negative values or values that does not sum to one. In the PAC terminology, the learning is said to be improper. As mentioned in (Balle et al., 2014; Gybels et al., 2014), this is a longstanding issue called the Negative Probability Problem (NPP). For some applications, the NPP is a major issue. For example, in reinforcement learning (Sutton & Barto, 1998), the value iteration algorithm can diverge when planning with an unbounded measure. Although some heuristics, that perform a local normalization, exist to recover probabilities on a finite set of events, they prevent the theoretical guarantees to hold. In addition, NPP prevents the use of MoM-based algorithms to initialize a local search with an iterative algorithm like EM. Some attempts try to perform a global normalization by projecting the learnt model onto the space of valid model parameters (Mossel & Roch, 2005; Gybels et al., 2014; Hsu et al., 2012; Anandkumar et al., 2012). While the resulting model is usable, these steps create an additional error and perform poorly in the experiments.

In this paper, we adopt the opposite approach. Instead of considering a more general class of automaton, we identify a subclass of models called Probabilistic Residual Finite Automaton (PRFA). We show that PRFA are PAC-learnable using a MoM-based algorithm that returns PFA and thus avoids the NPP. Although PRFA are strictly less general than PFA, their expressiveness is large enough to closely approximate many distributions used in practice. In addition, learnt models can serve as a good initialization to iterative algorithms that perform a local search in the more general class of PFA. The paper is organized as follows: in Section 2, we recall the definition of a PFA and the basic Spectral learning algorithm; in Section 3 we define PRFA and a provable learning algorithm, CH-PRFA, that runs in polynomial time; finally, we assess the performance of CH-PRFA on synthetic problems and a real large dataset that cannot be handled by traditional methods like EM.

2. Background

2.1. Probabilistic Finite Automaton

PFA are graphical models constrained to represent distributions over sequences of symbols. Let Σ be a set of symbols, also called an alphabet. We denote by $\Sigma^\star$, the set of all finite words made of symbols of Σ, including the empty word ε. Words of length k form the set Σ^k. Let u and $v \in \Sigma^\star$, uv is the concatenation of the two words and $u\Sigma^\star$ is the set of finite words starting by u. We are interested in capturing a distribution over $\Sigma^\star$. Let p be such a distribution, for a set of words $\mathcal{S}$, we define $p(\mathcal{S}) = \sum_{u\in\mathcal{S}} p(u)$, in particular we have that $p(\Sigma^\star) = 1$. In addition, for any word u we define $\overline{p}$ such that $\overline{p}(u) = p(u\Sigma^\star)$. Thus, $\overline{p}$ defines distributions over prefixes of fix length : $\forall n,\ \sum_{u\in\Sigma^n} \overline{p}(u) = 1$.

Some of these distributions can be modeled by graphical models called Probabilistic Finite Automaton (PFA).

Definition. *A PFA is a tuple* $\langle \Sigma, Q, \{A_o\}_{o\in\Sigma}, \boldsymbol{\alpha}_0, \boldsymbol{\alpha}_\infty \rangle$, *where* Σ *is an alphabet and* Q *is a finite set of states. Matrices* $A_o \in \mathbb{R}^{+|Q|\times|Q|}$ *contain the transition weights. The vectors* $\boldsymbol{\alpha}_\infty \in \mathbb{R}^{+|Q|}$ *and* $\boldsymbol{\alpha}_0 \in \mathbb{R}^{+|Q|}$ *contain respectively the terminal and initial weights. These weights should verify,*

$$\mathbf{1}^\top \boldsymbol{\alpha}_0 = 1 \qquad \boldsymbol{\alpha}_\infty + \sum_{o\in\Sigma} A_o \mathbf{1} = \mathbf{1} \tag{1}$$

A PFA realizes a distribution over $\Sigma^\star$, (Denis & Esposito, 2008), defined by

$$p(u) = p(o_1 \dots o_k) = \boldsymbol{\alpha}_0^\top A_u \boldsymbol{\alpha}_0 = \boldsymbol{\alpha}_0^\top A_{o_1} \dots A_{o_k} \boldsymbol{\alpha}_\infty. \tag{2}$$

Because of the constraints defined in Equation (1), the weights belong to $[0, 1]$ and can be viewed as probabilities over initial states, terminal states and transitions with symbol emission. For a word, we define a path as a sequence of states starting in an initial state, transiting from state to state, emitting symbols of the word in each state and exiting in a final state. The probability of a path is the product of the weights along the path including initial and final weights. Hence, the probability of a word is given by the sum of all paths probabilities, as written in Equation (2). A path (resp. word) with a positive probability is called an accepting path (resp. word).

2.2. Spectral Learning

Actually, PFA define a particular kind of Multiplicity Automaton (MA). A MA is also a tuple $\langle \Sigma, Q, \{A_o\}_{o\in\Sigma}, \boldsymbol{\alpha}_0, \boldsymbol{\alpha}_\infty \rangle$ but without any constraints on the weights, which can be negative and thus lose their probabilistic meaning. Thus, the function realized by a MA is not constrained to be a distribution. In the sequel, we call a Stochastic MA (SMA) a MA that realizes a distribution. In (Denis & Esposito, 2008), the authors showed that SMA are strictly more general than PFA and can be infinitely more compact (Esposito, 2004).

The Spectral algorithm presented in this section relies on the Hankel matrix representation of a function to learn a MA. Let $f : \Sigma^\star \to \mathbb{R}$ be a function, we define $H \in \mathbb{R}^{\Sigma^\star\times\Sigma^\star}$ the bi-infinite Hankel matrix whose rows and columns are indexed by $\Sigma^\star$ such that $H[u, v] = f(uv)$.

Hence, when f is a distribution, H contains occurrence probabilities that can be estimated from samples by counting occurrences of sequences. Let for all $o \in \Sigma$, $H_o \in \mathbb{R}^{\Sigma^\star \times \Sigma^\star}$ and $\mathbf{h} \in \mathbb{R}^{\Sigma^\star}$ be such that $H^o(u,v) = f(uov)$, $\mathbf{h}(u) = f(u)$. These vectors and matrices can be extracted from H. The Hankel representation lies at the heart of all MoM-based learning algorithms, because of the following fundamental theorem.

Theorem 1 (See (Carlyle & Paz, 1971)). *Let f be a function realized by a MA with n states, then $\mathrm{rank}(H) \leq n$. Conversely, if the Hankel matrix H of a function $f : \Sigma^\star \to \mathbb{R}$ has a finite rank n, then f can be realized by a MA with exactly n states but not less.*

For a MA with n states, observe that $H[u,v] = (\boldsymbol{\alpha}_0^\top A_u)(A_v \boldsymbol{\alpha}_\infty)$. Let $P \in K^{\Sigma^\star \times n}$ and $S \in K^{n \times \Sigma^\star}$ be matrices defined as follows,

$$P = ((\boldsymbol{\alpha}_0^\top A_u)^\top)_{u \in \Sigma^\star}^\top, \qquad S = (A_v \boldsymbol{\alpha}_\infty)_{v \in \Sigma^\star},$$

then $H = PS$. Moreover, we have that,

$$H_o = P A_o S, \quad \mathbf{h}^\top = \boldsymbol{\alpha}_0^\top S, \quad \mathbf{h} = P \boldsymbol{\alpha}_\infty. \tag{3}$$

So, the MA parameters can be recovered by solving Equation (3). Hopefully, we do not need to consider the bi-infinite Hankel matrix to recover the underlying MA. Given a basis $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ of prefixes and suffixes, we denote by $H_\mathcal{B}$ the sub-block of H. Similarly, $H^o_\mathcal{B}$ is a sub-block of H^o and $\mathbf{h}_\mathcal{P}$ and $\mathbf{h}_\mathcal{S}$ are sub-blocks of $\mathbf{h}$. We say, that a basis $\mathcal{B}$ is *complete* if $H_\mathcal{B}$ has the same rank than H. In (Balle, 2013), the author shows that if $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ is a complete basis, by also defining P over $\mathcal{P}$, S over $\mathcal{S}$, we can recover a MA using Equation (3). Several methods are proposed in the literature to build a complete basis from data. In the experiments, we used the most frequent prefixes and suffixes.

Once a basis is chosen, the Spectral algorithm first estimates $\hat{H}_\mathcal{B}$. Then, it approximates $\hat{H}_\mathcal{B}$ with a low dimensional factorized form $\hat{H}_\mathcal{B} \approx \hat{U}\hat{D}\hat{V}^\top$ through a truncated Singular Value Decomposition (SVD). Finally, setting $\hat{P} = \hat{U}\hat{D}$ and $\hat{S} = \hat{V}^\top$, the algorithm solves Equation (3), through linear regression. Because of the properties of the SVD, this leads to the following equations :

$$\begin{aligned}
\hat{A}_o &= \hat{D}^{-1}\hat{U}^\top \hat{H}^o_\mathcal{B} \hat{V}\,, \\
\hat{\boldsymbol{\alpha}}_0^\top &= \hat{\mathbf{h}}_\mathcal{S}^\top \hat{V}\,, \\
\hat{\boldsymbol{\alpha}}_\infty &= \hat{D}^{-1}\hat{U}^\top \hat{\mathbf{h}}_\mathcal{P}.
\end{aligned}$$

Although the Spectral algorithm can return a MA arbitrary close to a SMA that realizes the target distribution, it does not ensure that the returned MA will be a SMA (and so a PFA). This causes the NPP explained in introduction.

Recalling that our goal is to learn proper distributions, one would like to add constraints to ensure that the MA learned by regression realizes a proper distribution. Unfortunately, this would require two things : 1) the non-negativity of series for any word ($\forall \in \Sigma^\star$, $p(u) \geq 0$), 2) the convergence of the series to one ($\sum_{u \in \Sigma^\star} p(u) = 1$). Although 2) can be checked in polynomial time, 1) requires adding an infinite set of constraints during the linear regression step. That is why, in general, verifying if a MA is a SMA is undecidable (Denis & Esposito, 2008). Actually, it has been shown in (Esposito, 2004) that no algorithm can learn a SMA in the limit of an infinite number of samples with probability 1.

So, in a first attempt we could restrict ourselves to the learning of PFA that are identifiable in the limit with probability 1 (Denis & Esposito, 2004). However, in (Kearns et al., 1994), the authors showed that PFA are not PAC-learnable by reduction to the learning of noisy parity functions, which is supposed to be difficult. Note that from (Abe & Warmuth, 1990), we know that this negative result comes from the computational complexity, as only a polynomial number of samples could suffice. Thus, in this work we will focus on a smaller, but still rich, set of automata, called Probabilistic Residual Finite Automata (PRFA) that have been introduced in (Denis & Esposito, 2008).

3. Probabilistic Residual Finite Automata

This section defines a particular kind of MA called Probabilistic Residual Finite Automata that realizes distributions. First, for any word u, we define the linear operator $\dot{u}$ on functions of $\mathbb{R}^{\Sigma^\star}$ such that $\forall v \in \Sigma^\star$, $\dot{u}p(v) = p(uv)$. Then, for any distribution p, we denote, for each word u such that $\overline{p}(u) > 0$, p_u the conditional distribution defined by $p_u = \frac{\dot{u}p}{\overline{p}(u)}$. In addition, for a PFA $\left\langle \Sigma, Q, \{A_o\}_{o \in \Sigma}, \boldsymbol{\alpha}_0, \boldsymbol{\alpha}_\infty \right\rangle$ realizing a distribution p, we denote, for all $q \in Q$, p_q the distribution defined by $p_q(u) = \mathbf{1}_q^\top A_u \boldsymbol{\alpha}_\infty$. Thus, $p_q(u)$ is the probability of observing u starting from the state q. Similarly, $p_u(v)$ is the probability to observe v after u.

Definition. *A PRFA is a PFA $(\Sigma, Q, \boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ such that for all state $q \in Q$, there exists a word $u \in \Sigma^\star$ such that $\overline{p}(u) > 0$ and $p_q = p_u$.*

In particular, a PFA such that, for all state q there exists at least one prefix of an accepted word that ends only in state q, is a PRFA. In addition, if the PFA is reduced (there is no PFA with strictly less state realizing the same language), the converse is true. Note that as a PRFA is a PFA, it realizes a distribution, which we denote by p. In addition, for all $q \in Q$, p_q is also a distribution (see (Denis & Esposito, 2008)).

From that definition, we see that PRFA are more general than PFA with deterministic transition (PDFA). In fact, PRFA are strictly more general than PDFA but strictly less general than PFA (Esposito, 2004). Similarly, the equiva-

lent of PRFA for stochastic processes lies between HMMs and any finite order Markov chains.

Now, we show how the existence of a set of words verifying some properties characterizes a PRFA. This equivalence allows us to design a learning algorithm.

Proposition 2. *Let p be a distribution, if there exists a set of words $\mathcal{R}$ and two associated sets of non-negative reals $\{a^v_{u,o}\}_{u,v\in\mathcal{R},o\in\Sigma}$ and $\{a^v_\varepsilon\}_{v\in\mathcal{R}}$, such that $\forall u\in\mathcal{R}, \overline{p}(u) > 0$ and,*

$$\forall u\in\mathcal{R},\ o\in\Sigma,\ \dot{o}p_u = \sum_{v\in\mathcal{R}} a^v_{u,o}p_v \text{ and } p = \sum_{v\in\mathcal{R}} a^v_\varepsilon p_v, \tag{4}$$

then, $\langle \Sigma, Q, \{A_o\}_{o\in\Sigma}, \boldsymbol{\alpha}_0, \boldsymbol{\alpha}_\infty\rangle$ defines a PRFA realizing p, where $Q=\mathcal{R}$,

$$\boldsymbol{\alpha}_0^\top = (a^u_\varepsilon)^\top_{u\in\mathcal{R}}, \tag{5}$$

$$\boldsymbol{\alpha}_\infty = (p_u(\varepsilon))_{u\in\mathcal{R}}, \tag{6}$$

$$\forall u,v\in\mathcal{R},\ A_o[u,v] = a^v_{u,o}. \tag{7}$$

Proof. First, we show by induction on the length of u that $\forall u\in\Sigma^\star, (p_v(u))_{v\in\mathcal{R}} = A_u\boldsymbol{\alpha}_\infty$. By the definition of $\boldsymbol{\alpha}_\infty$, the property is verified for $u=\varepsilon$. Assume the property is true for $u\in\Sigma^{\leq n}$, then for any $o\in\Sigma$, let $v=ou$ and we have,

$$\begin{aligned}
A_v\boldsymbol{\alpha}_\infty &= A_oA_u\boldsymbol{\alpha}_\infty \\
&= A_o\left(p_w(u)\right)_{w\in\mathcal{R}} \\
&= \left(\sum_{w\in\mathcal{R}} a^w_{w',o}p_w(u)\right)_{w'\in\mathcal{R}} && \text{(by Equation (5))} \\
&= \left(\dot{o}p_{w'}(u)\right)_{w'\in\mathcal{R}} && \text{(by Equation (4))} \\
&= \left(p_w(v)\right)_{w\in\mathcal{R}}.
\end{aligned}$$

Now, for all $u\in\Sigma^\star$, with Equations (4) and (5), we have

$$\boldsymbol{\alpha}_\infty^\top A_u\boldsymbol{\alpha}_\infty = \sum_{w\in\mathcal{R}} a^w_\varepsilon p_w(u) = p(u).$$

It remains to show that $\langle \Sigma, Q, \{A_o\}_{o\in\Sigma}, \boldsymbol{\alpha}_0, \boldsymbol{\alpha}_\infty\rangle$ defines a PRFA, meaning that it satisfies Equation (1) and

$$\forall q\in Q,\ \exists u\in\Sigma^\star,\ (\overline{p}(u)>0)\wedge(p_q = p_u). \tag{8}$$

From Equation (4), we have $p=\sum_{u\in\mathcal{R}} a^u_\varepsilon p_u$. In particular $p(\Sigma^\star) = \sum_{u\in\mathcal{R}} a^u_\varepsilon p_u(\Sigma^\star)$ and $p(\Sigma^\star) = p_u(\Sigma^\star) = 1$ for all $u\in\mathcal{R}$ implies that $\sum_{u\in\mathcal{R}} a^u_\varepsilon = 1$. Finally, from Equation (4), we have for all $u\in\mathcal{R}$ that

$$\begin{aligned}
\dot{o}p_u(\Sigma^\star) &= \sum_{v\in\mathcal{R}} a^v_{u,o}p_v(\Sigma^\star), \\
\sum_{o\in\Sigma} p_u(o\Sigma^\star) &= \sum_{o\in\Sigma}\sum_{v\in\mathcal{R}} a^v_{u,o}, \\
p_u(\Sigma\Sigma^\star) &= \sum_{o\in\Sigma}\sum_{v\in\mathcal{R}} a^v_{u,o}.
\end{aligned}$$

As p_u is a distribution, $p_u(\Sigma^\star) = p_u(\varepsilon)+p_u(\Sigma\Sigma^\star)$. So, we obtain that $\boldsymbol{\alpha}_\infty + A_\Sigma\mathbf{1} = \mathbf{1}$. Hence, $(\boldsymbol{\alpha}_0, A, \boldsymbol{\alpha}_\infty)$ satisfies Equation (1). We showed that for any word $v\in\Sigma^\star$,

$$\left(p_q(v)\right)_{q\in Q} = A_v\boldsymbol{\alpha}_\infty = \left(p_u(v)\right)_{u\in\mathcal{R}}.$$

So by taking $Q=\mathcal{R}$, we have that Equation (8) is satisfied and $\langle \Sigma, Q, \{A_o\}_{o\in\Sigma}, \boldsymbol{\alpha}_0, \boldsymbol{\alpha}_\infty\rangle$ is a PRFA. □

In Proposition 3, we show the converse, i.e. the existence of the $\mathcal{R}$, $\{a^v_{u,o}\}_{u,v\in\mathcal{R},o\in\Sigma}$ and $\{a^v_\varepsilon\}_{v\in\mathcal{R}}$ for a PRFA. In fact, the existence of coefficients satisfying Equation (4) can be reformulated using the notion of conical hull. We denote by $\mathrm{coni}(E)$ the set defined by,

$$\mathrm{coni}(E) = \{\sum_{e\in E}\alpha_e e | \alpha_e\in\mathbb{R}^+\}.$$

Proposition 3. *Let $\langle \Sigma, Q, \{A_o\}_{o\in\Sigma}, \boldsymbol{\alpha}_0, \boldsymbol{\alpha}_\infty\rangle$ be a PRFA and p the distribution it realizes, then there exists a set of words $\mathcal{R}$ such that*

$$\begin{gathered}
\forall u\in\mathcal{R}, \overline{p}(u) > 0, \\
p\in\mathrm{coni}\{p_u | u\in\mathcal{R}\}, \\
\forall u\in\mathcal{R},\ o\in\Sigma,\ \dot{o}p_u\in\mathrm{coni}\{p_v | v\in\mathcal{R}\}.
\end{gathered}$$

Proof. From the definition of a PRFA given Equation (8), there exists a finite set of words $\mathcal{R}$ of size bounded by $|Q|$, the number of state, such that

$$\forall q\in Q,\ \exists u\in\mathcal{R},\ (\overline{p}(u)>0)\wedge(p_q = p_u).$$

To prove the existence of $\{a^v_{u,o}\}_{u,v\in\mathcal{R},o\in\Sigma}$ and $\{a^v_\varepsilon\}_{v\in\mathcal{R}}$, we write that for any $w\in\Sigma^\star$,

$$p(w) = \boldsymbol{\alpha}_0^\top A_w\boldsymbol{\alpha}_\infty = \boldsymbol{\alpha}_0^\top\left(p_q(w)\right)_{q\in Q} = \boldsymbol{\alpha}_0^\top\left(p_u(w)\right)_{u\in\mathcal{R}}.$$

As for a PRFA, $\boldsymbol{\alpha}_0$ is a vector of non-negative coefficients, we have that $p\in\mathrm{coni}\{p_u | u\in\mathcal{R}\}$. Similarly, for all $u\in\mathcal{R}$, $o\in\Sigma$, we have for any $w\in\Sigma^\star$

$$\begin{aligned}
\dot{o}p(w) = \boldsymbol{\alpha}_0^\top A_{ow}\boldsymbol{\alpha}_\infty &= \boldsymbol{\alpha}_0^\top A_o\left(p_q(w)\right)_{q\in Q} \\
&= \boldsymbol{\alpha}_0^\top A_o\left(p_u(w)\right)_{u\in\mathcal{R}}.
\end{aligned}$$

As $\boldsymbol{\alpha}_0$ and A_o are non-negative, $\boldsymbol{\alpha}_0^\top A_o$ is a vector with non-negative coefficients and we have that $\forall u\in\mathcal{R}$, $o\in\Sigma$, $\dot{o}p_u\in\mathrm{coni}\{p_v | v\in\mathcal{R}\}$. □

4. Learning PRFA

As in the Spectral algorithm, our method assumes that a complete basis $\mathcal{B}=(\mathcal{P},\mathcal{S})$ is provided. To simplify the discussion, we will also assume that the empty word ε is included as a prefix and a suffix in the basis. For convenience, we denote $\mathbf{1}_\varepsilon$ a vector on $\mathcal{P}$ or $\mathcal{S}$, depending on the

context, filled with zeros but a one at the index of ε. In addition, this basis must contain the prefixes and the suffixes allowing the identification of a set $\mathcal{R}$ generating a conical hull containing p and, for all $u \in \mathcal{R}$, $\dot{o}p_u$. In the sequel, we denote $\mathbf{p}$ (resp. $\mathbf{p}_u$, $\dot{o}\mathbf{p}_u$), the vector representation of p (resp. p_u, $\dot{o}p_u$) on the basis of suffixes $\mathcal{S}$. Thus, in addition to be complete, the basis $\mathcal{B}$ must be residual.

Definition (Residual basis)**.** *A basis $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ is residual if the conical hull* $\operatorname{coni}\{p_u | u \in \Sigma^\star, \overline{p}(u) > 0\}$ *projected on $\mathcal{S}$ coincide with* $\operatorname{coni}\{\mathbf{p}_u | u \in \mathcal{P}, \mathbf{p}_u > 0\}$.

In addition, we assume the Algorithm 1 is provided with the minimal dimension d of PRFA realizing p. Hence, as the basis is complete and residual, we have that the hypothesis in Proposition 2 are satisfied, i.e. there exists $\mathcal{R} \subset \mathcal{P}$ such that $\forall u \in \mathcal{R}, \overline{p}(u) > 0$, $\mathbf{p} \in \operatorname{coni}\{\mathbf{p}_u | u \in \mathcal{R}\}$ and $\forall u \in \mathcal{R}$, $o \in \Sigma$, $\dot{o}\mathbf{p}_u \in \operatorname{coni}\{\mathbf{p}_v | v \in \mathcal{R}\}$.

The CH-PRFA algorithm works by first estimating the $\mathbf{p}_u$ with $u \in \mathcal{P}$. Then, it finds the set $\hat{\mathcal{R}}$. We will see this step can be solved using near-separable Non-negative Matrix Factorization (NMF). To identify $\hat{\mathcal{R}}$, instead of using $\{\hat{\mathbf{p}}_u | u \in \mathcal{P}\}$, we used $\{\hat{\mathbf{d}}_u | u \in \mathcal{P}\}$, where $\hat{\mathbf{d}}_u$ is defined in Algorithm 1 because it improves the robustness and helps in the proof of Theorem 4. Finally, the parameters of a PFA are retrieved through linear regressions. Because of estimation errors, we need to add non-negativity and linear constraints to ensure the parameters defines a PFA. Hence, CH-PRFA returns a PFA but not necessarily a PRFA. In the PAC terminology, our algorithm is improper like the Spectral one. However, it is inconsequential as a PFA realizes a proper distribution. In contrast, we recall that the Spectral learning algorithm returns a MA that does not necessarily realizes a proper distribution. In addition, our algorithm is proper in limit whereas the Spectral algorithm is not.

In the literature, many algorithms for NMF have been proposed. Although in its general form NMF is NP-Hard and ill-posed (Gillis, 2014), in the near-separable case the solution is unique and can be found in $O(k\,|\mathcal{S}|\,|\mathcal{P}|)$ steps. State-of-the-art algorithms for near-separable NMF comes with convergence guarantees and robustness analysis. In our experiment, the Successive Projection Algorithm (SPA), analyzed in (Gillis & Vavasis, 2014), gave good results. In addition, SPA is very efficient and can be easily distributed.

In Algorithm 1, the minimization problems can be cast as quadratic optimization problems under linear constraints with convex costs. This kind of problem can be solved in polynomial time using a solver like Mosek (MOSEK, 2015). As the cost is convex, solvers converge to a stationary point. In addition, all stationary points have the same cost and are optimal (Lőtstedt, 1983). To get a unique solution, it is possible to look for the minimal norm solution, as a Moore-Pseudo inverse does.

Algorithm 1 CH-PRFA

Input: A targeted dimension d, a separable complete basis $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ and a training set.

Output: A PFA $\left\langle \Sigma, \hat{\mathcal{R}}, \left\{\hat{A}_o\right\}_{o\in\Sigma}, \hat{\boldsymbol{\alpha}}_0, \hat{\boldsymbol{\alpha}}_\infty \right\rangle$.

for $u \in \hat{\mathcal{P}}$ **do**
  Estimate $\hat{\mathbf{p}}_u$ and $\dot{o}\hat{\mathbf{p}}_u$ from the training set.
  $\hat{\mathbf{d}}_u \leftarrow \left(\hat{\mathbf{p}}_u^\top \; \dot{o}_1\hat{\mathbf{p}}_u^\top \; \dots \; \dot{o}_{|\Sigma|}\hat{\mathbf{p}}_u^\top\right)^\top$
end for
Find $\hat{\mathcal{R}}$ a subset of d prefixes of $\mathcal{P}$ such that $\forall u \in \hat{\mathcal{R}}, \hat{\mathbf{d}}_u > 0$ and $\hat{\mathbf{d}} \in \operatorname{coni}\{\hat{\mathbf{d}}_u | u \in \hat{\mathcal{R}}\}$.
for $u \in \hat{\mathcal{R}}$ **do**

$$\{\hat{a}^v_{u,o}\} \leftarrow \operatorname*{argmin}_{\{a^v_{u,o}\}} \sum_{o\in\Sigma} \left\| \dot{o}\hat{\mathbf{p}}_u - \sum_{v\in\hat{\mathcal{R}}} a^v_{u,o}\hat{\mathbf{p}}_v \right\|_2$$
$$\text{s.t.} \sum_{v\in\hat{\mathcal{R}},o\in\Sigma} a^v_{u,o}\dot{o} = 1 - \hat{\mathbf{p}}_u\mathbf{1}_\epsilon \text{ and } a^v_{u,o} \geq 0\,.$$

end for

$$\{\hat{a}^u_\varepsilon\} \leftarrow \operatorname*{argmin}_{\{a^u_\varepsilon\}} \left\| \hat{\mathbf{p}} - \sum_{u\in\hat{\mathcal{R}}} a^u_\varepsilon \hat{\mathbf{p}}_u \right\|_2$$
$$\text{s.t.} \sum_{u\in\hat{\mathcal{R}}} a^u_\varepsilon = 1 \text{ and } a^u_\varepsilon \geq 0.$$

$$\hat{\boldsymbol{\alpha}}_0^\top \leftarrow (\hat{a}^u_\varepsilon)^\top_{u\in\hat{\mathcal{R}}}, \qquad \hat{\boldsymbol{\alpha}}_\infty \leftarrow (\hat{\mathbf{p}}_u\mathbf{1}_\epsilon)_{u\in\hat{\mathcal{R}}}.$$

for $o \in \Sigma$ **do**
  $\hat{A}_o \leftarrow \left(\hat{a}^v_{u,o}\right)_{u,v\in\hat{\mathcal{R}}}$.
end for

Using the perturbation analysis of SPA (Gillis & Vavasis, 2014) and the one of quadratic optimization (Lőtstedt, 1983), we show the following non-asymptotic bound.

Theorem 4. *Let p be a distribution realized by a minimal PRFA of size d, $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ be a complete and residual basis, we denote by σ_d the d-th largest singular values of $(\mathbf{p}_u(v))_{u\in\mathcal{R}}$. Let $\mathcal{D}$ be a training set of words generated by p, we denote by n the number of time the least occurring prefix of $\mathcal{P}$ appears in $\mathcal{D}$ ($n = \min_{u\in\mathcal{P}} |\{\exists v \in \Sigma^\star | uv \in \mathcal{D}\}|$). For all $0 < \delta < 1$, there exists a constant K such that, for all $t > 0$, $\epsilon > 0$, with probability $1 - \delta$, if*

$$n \geq K \frac{t^4 d^4 \,|\Sigma|}{\epsilon^2 \sigma_d^{10}} \log\left(\frac{|\mathcal{P}|}{\delta}\right),$$

CH-PRFA returns a PFA realizing a proper distribution $\hat{p}$ such that

$$\sum_{u\in\Sigma^{\leq t}} |\hat{p}(u) - p(u)| \leq \epsilon.$$

Proof. In the supplementary material. □

Now, we compare this result to previous bounds on the Spectral algorithm. First our bound depends on n, instead

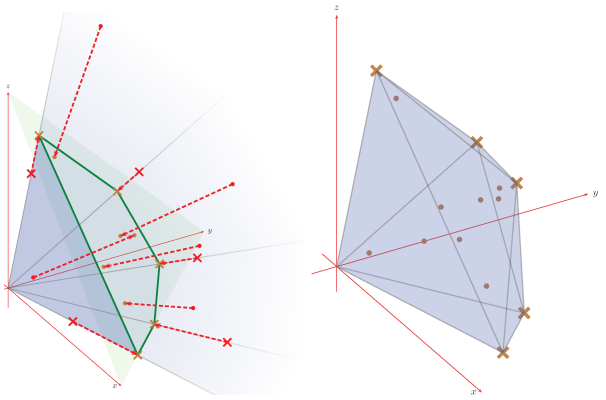

Figure 1. On the left, a conical hull of a set of points and their projection onto the simplex. On the right, a convex hull of a set of points.

of $N = |\mathcal{D}|$. Using Hoeffding inequality, we could obtain a bound on N depending on the inverse of the probability of the least frequent prefix of $\mathcal{P}$. This dependence comes from the use of conditional distributions (the $\mathbf{p}_u$) instead of joint distributions. Indeed, the Hankel matrix used in Spectral contains joint distributions. Removing this dependency on the prefix set seems possible by changing how the NMF is computed. This direction will be explored in future researches. Then, we address the convergence speed. First, the dependency on $\log(|\mathcal{P}|)$ could be removed using recent dimension-free concentration bounds on Hankel matrices (Denis et al., 2014). Secondly, although the term $|\Sigma|$ is better than $|\Sigma|^2$ in the classical error bounds for Spectral (Hsu et al., 2012; Balle, 2013), the results from (Foster et al., 2012) on HMMs suggest that the error bound could be independent of $|\Sigma|$. Third, the convergence speed in $O(\epsilon^{-2}\log(\delta^{-1}))$ comes directly from concentration results and seems optimal. Finally, the required number of sample depends on d^4 which is worst than in the bounds for the Spectral algorithm. This strong dependency on d comes from the constraints in the optimization problems. Finally, the high polynomial order of σ_d^{-1} in the bound is a direct consequence of SPA.

5. Experimental results

5.1. Near-Separable NMF

In this Section, we give more details on the identification of a conical hull containing a set of vectors. This problem has often been addressed as a NMF problem, where a non-negative matrix as to be decomposed in two non-negative matrices of reduced dimensions. One of these contains the vectors supporting the conical hull and the other the conical combinations to recover all the vectors of the original matrix. This problem, in its general form, is ill-posed and NP-Hard (Gillis, 2014) and algorithms often rely on alternated optimization that converge only to a local optimum. A decade ago (Donoho & Stodden, 2003), a sufficient condition, called separability, have been identified to ensure the uniqueness of the solution. Geometrically, separability implies that the vectors supporting the conical hull are contained in the original matrix. Since, many algorithms relying on different additional assumptions have been proposed to solve NMF for separable matrices.

In particular, the Successive Projection Algorithm used both in the finite sample analysis and in the experiments, identifies recursively the supporting vectors among the column of the original matrix. It assumes that the matrix formed with the supporting vectors, $(\mathbf{p}_u(v))_{u\in\mathcal{R}}$ in our case, has full rank. Although the full rank assumption is not generally true, as we work with empirical estimate it is satisfied with probability 1. Moreover, experimental results does not seem to suffer from that assumption. Additionally, SPA assumes the supporting vectors form a convex hull instead of a conical hull. This assumption can be made without loss of generality as columns from the original matrix can be normalized to belong to the simplex. In our case, this assumption is already satisfied because of the constraints given in Equation (1). However, we will explain why a direct application of the SPA algorithm constrains which prefixes can be included in the basis.

As suggested by the finite sample analysis and experiences, incorporating prefixes with low occurring probabilities in the basis degrades strongly the performances. In fact, the robustness of SPA depends on the maximum error in norm made on $\mathbf{p}_u$. However, probabilities conditioned on rare prefixes are not well estimated. Therefore, taking only the most frequent prefixes is needed to achieve good results. In order to fix this issue, we propose another way to use the SPA algorithm. By rescaling the vectors, we can obtain the same amount of uncertainty for all prefixes. So, instead of working with estimates of $\mathbf{p}_u = \left(\frac{p(uv)}{\overline{p}(u)}\right)_{v\in\mathcal{S}}$, we use the vectors $\mathbf{q}_u = (p(uv))_{v\in\mathcal{S}}$ (where $u \in \mathcal{P}$). In other words, the SPA algorithm is executed on the same Hankel matrix than the Spectral algorithm. The Figure 1 shows the differences between the conical hull and the convex hull as well as the effect of normalizing the vectors to the simplex.

One can wonder if the algorithm is still valid because, even with the true vectors, the result of SPA will differ depending on whether we use $\mathbf{p}_u$ or $\mathbf{q}_u$. Let $\mathcal{R}$ be the set of prefixes identified by SPA using $\mathbf{p}_u$. Let $\mathcal{Q}$ be the set of prefixes identified by SPA using $\mathbf{q}_u$. The algorithm is still valid if only a finite number of vectors $\mathbf{q}_u$ lies outside the convex hull generated by $\{\mathbf{q}_u | u \in \mathcal{R}\}$. Thus, as long as the model dimension is large enough $\mathcal{Q}$ will contain $\mathcal{R}$. Including more prefixes than needed is inconsequential for the remaining part of the algorithm. So, using $\mathbf{q}_u$ instead

Table 1. Average WER on twelve problems of PAutomaC.

RANK	ALGORITHM	WER
1.	NNSPECTRAL	64.618
2.	CH-PRFA+BW	65.160
3.	SPECTRAL	65.529
4.	BW	66.482
5.	CH-PRFA	67.141
6.	CO	71.467
7.	TENSOR+BW	73.544
8.	TENSOR	77.433

Table 2. Average perplexity on twelve problems of PAutomaC.

RANK	ALGORITHM	PERPLEXITY
1.	NNSPECTRAL	30.364
2.	CH-PRFA+BW	30.468
3.	CH-PRFA	30.603
4.	CO	35.641
5.	BW	35.886
6.	SPECTRAL	40.210
7.	TENSOR+BW	47.655
8.	TENSOR	54.000

of $\mathbf{p}_u$ leave the validity CH-PRFA unchanged. In the experiments, we used this variation of SPA as it increased the performances without increasing models sizes.

5.2. PAutomaC Challenge

The Probabilistic Automata learning Competition (PAutomaC) deals with the problem of learning probabilistic distributions from strings drawn from finite-state automata. From the 48 problems available, we have selected the same 12 problems than in (Balle et al., 2014), to provide a fair comparison with other algorithms. The generating model can be of three kinds: PFA, HMMs or PDFA. A detailed description of each problem can be found in (Verwer et al., 2012). We compared CH-PRFA and CH-PRFA+BW (BW initialized with CH-PRFA) to Baum-Welch (BW) with 3 random restarts of 3 iterations (then the best run is continued for a maximum of 30 iterations) and other MoM-based algorithms : CO (Balle et al., 2012), Tensor (Anandkumar et al., 2012), NNSpectral (Glaude et al., 2015) and Spectral with variance normalization (Cohen et al., 2013). In the experiments, negative values outputted by the algorithms are set to zero, then we normalize to obtain probabilities. These MoM-based algorithms have been trained using statistics on sequences and subsequences as proposed in (Balle, 2013). The best result is then selected. For all MoM-based algorithm, we used basis sizes varying between 50 and 10000 (except for CO where the computation time limits the basis size to 200 and NNSpectral to 500). For BW, we stopped the iterations if after 4 iteration the score was not improving. To obtain probability-like values from MoM-based algorithms we used several tricks. For Spectral and CO, we zeroed negative values and normalized. For NNSpectral, we just normalized. For Tensor, we projected the transition and observation matrix onto the simplex, as described in (Balle et al., 2014). Finally, we assessed the quality of the learned distribution p by the perplexity. It corresponds to the average number of bits needed to represent a word using the optimal code $p^\star$.

$$\text{Perplexity}(\mathcal{M}) = 2^{-\sum_{u\in\mathcal{T}} p^\star(u)\log(p_\mathcal{M}(u))}.$$

We also measured the quality of p by computing the Word Error Rate (WER) which is the average number of incorrect predictions of the next symbols given the past ones. A grid search was performed to find the optimal dimension and basis size for each of the performance metric. On Tables 1 and 2, we ranked the algorithm according to their average performances on the twelve problems. For the WER, the average corresponds to the mean. For the perplexity, the average corresponds to $2^{-\sum_{i=1}^{12}\sum_{u\in\mathcal{T}} p_i^\star(u)\log(p_{\mathcal{M}_i}(u))}$.

5.3. Wikipedia

We also evaluated CH-PRFA and CH-PRFA+BW on raw text extracted from English Wikipedia pages. The training set is made of chunks of sequences of 250 characters randomly extracted from the 2GB corpus used in (Sutskever et al., 2011). Each character stands for a symbol. We restricted ourself to 85 different symbols. For the training phase, we used a small set of 500 chunks and a medium one of 50000 characters. For testing, an independent set of 5000 chunks has been used. The algorithms are the same than the ones evaluated for PAutomaC but slightly modified to fit a stochastic process. For example, we used the suffix-history algorithm (Wolfe et al., 2005) to estimate the occurrence probabilities in the Hankel matrices. For CH-PRFA, we changed the constraints defined in Equation (1) and used during the optimization problems to be the ones of a stochastic process. As the results of CO and Tensor were very poor (between 0.05 and 0.10 for the likelihood), we did not report them for clarity. The time taken by BW were excessive (more than a day in comparison to the tens of minutes needed by the others) for model sizes above 60 on the small training set and for all dimensions on the medium one. Performance is measured by the average likelihood of the next observation given the past. More precisely, on chunk of 250 characters, the 50 firsts served to initialize the belief, then the next observations are predicted based on the characters observed so far. The likelihood of all these predictions is then averaged over observations and chunks. We also computed the number of bits per character (BPC) by averaging $\log(\mathbb{P}(o_{t+1}|o_{1:t}))$ over sequences $o_{1:t}$ of symbols.

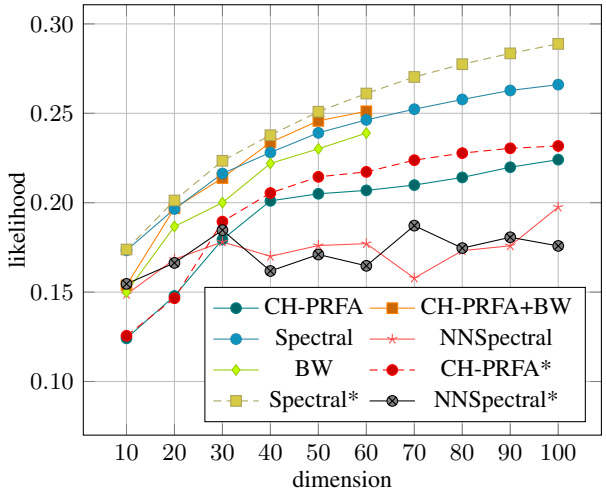

Figure 2. Likelihood on Wikipedia (the higher is the better). Asterisks denote scores on the medium training set.

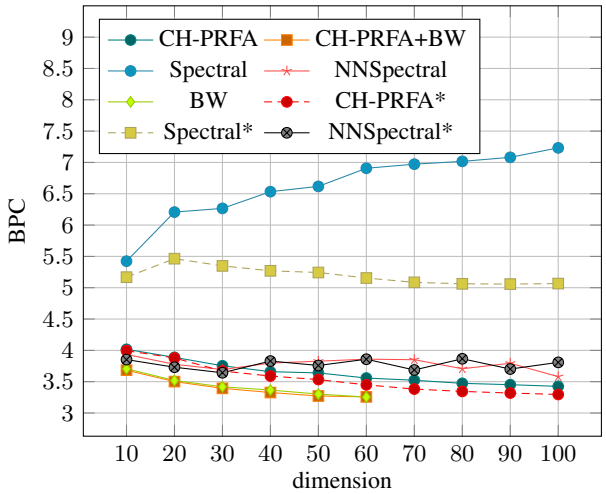

Figure 3. BPC on Wikipedia (the lower is the better). Asterisks denote scores on the medium training set.

6. Discussion

On PAutomaC, the top performer is NNSpectral both for the perplexity and the WER. CH-PRFA performed almost as good as NNSpectral considering the perplexity. The mean WER of CH-PRFA is slightly less good than NNSpectral, Spectral and BW. Although CH-PRFA is not the top performer, it has PAC-style guarantees in contrast to NNSpectral and BW and is much faster as shown on Figure 4. Finally, in contrast to Spectral, CH-PRFA does not suffer from the NPP.

On Wikipedia, for the likelihood, the top performer is Spectral. We believe that its good scores come from the fact that MA are more strictly more general and more compact than PFA and so PRFA. On a training set like Wikipedia, it can be a big asset as a natural language model is likely to be very complex. The mixed performances of CH-PRFA can be explained by the lack of expressiveness of PRFA. The lack of expressiveness of PRFA is then filled using BW to improve the model quality as BW learns a PFA. Hence, when trained on the small set, CH-PRFA+BW achieves the best performances and beats Spectral. However, on the medium set the BW algorithm takes too much time to be a decent alternative. For the BPC, results are quite surprising as Spectral is the least performer. In fact, the BPC gives more importance to rare events than the conditional likelihood. So, Spectral predicts better frequent events than rare events. Indeed, small probabilities associated to rare events are likely to be a sum of products of small parameters of the MA. As parameters are not constrained to be non-negative, a small error can flip the sign of small parameters which in turn leads to large errors. That is why, NNSpectral and CH-PRFA performed better than Spectral for the BPC.

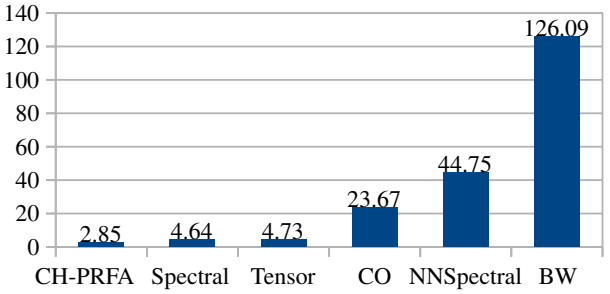

Figure 4. Mean learning time on twelve problems of PAutomaC.

Finally, using two sets of different sizes shows that BW do not scale well. The running time of BW between the two sets has changed from few minutes to at least a day, whereas the one of CH-PRFA has only increased by few seconds.

7. Conclusions

In this paper, we proposed a new algorithm based on a near-separable NMF and constraint quadratic optimization that can learn in polynomial time a PRFA from the distribution p it realizes. In addition, even if p is not realized by a PRFA or is empirically estimated, our algorithm returns a PFA that realizes a proper distribution closed to the true distribution. We established PAC-style bounds that allowed us to tweak the NMF to achieve better results. Then, we empirically demonstrated its good performances in comparison to other MoM-based algorithms, and, its scalability in comparison to BW. Finally, experiments have shown that initializing BW with CH-PRFA can substantially improve the performances of BW. For future works, extending Algorithm 1 to handle controlled processes would allow designing consistent reinforcement learning algorithms for non-Markovian environments.

References

Abe, Naoki and Warmuth, Manfred K. On the computational complexity of approximating distributions by probabilistic automata. In *Proceedings of the Third Annual Workshop on Computational Learning Theory, COLT 1990, University of Rochester, Rochester, NY, USA, August 6-8, 1990.*, pp. 52–66, 1990.

Anandkumar, Anima, Ge, Rong, Hsu, Daniel, Kakade, Sham M., and Telgarsky, Matus. Tensor decompositions for learning latent variable models. *CoRR*, abs/1210.7559, 2012.

Balle, Borja. *Learning finite-state machines: statistical and algorithmic aspects*. PhD thesis, Universitat Politècnica de Catalunya, 2013.

Balle, Borja, Quattoni, Ariadna, and Carreras, Xavier. Local loss optimization in operator models: A new insight into spectral learning. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*, 2012.

Balle, Borja, Hamilton, William L., and Pineau, Joelle. Methods of moments for learning stochastic languages: Unified presentation and empirical comparison. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pp. 1386–1394, 2014.

Carlyle, Jack W. and Paz, Azaria. Realizations by stochastic finite automata. *J. Comput. Syst. Sci.*, 5(1):26–40, 1971.

Cohen, Shay B., Stratos, Karl, Collins, Michael, Foster, Dean P., and Ungar, Lyle H. Experiments with spectral learning of latent-variable PCfgs. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*, pp. 148–157, 2013.

Denis, François and Esposito, Yann. Learning classes of probabilistic automata. In *Learning Theory*, pp. 124–139. Springer, 2004.

Denis, François and Esposito, Yann. On rational stochastic languages. *Fundam. Inform.*, 86(1-2):41–77, 2008.

Denis, François, Gybels, Mattias, and Habrard, Amaury. Dimension-free concentration bounds on Hankel matrices for spectral learning. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pp. 449–457, 2014.

Donoho, David L. and Stodden, Victoria. When does nonnegative matrix factorization give a correct decomposition into parts? In *Advances in Neural Information Processing Systems 16 [Neural Information Processing Systems, NIPS 2003, December 8-13, 2003, Vancouver and Whistler, British Columbia, Canada]*, pp. 1141–1148, 2003.

Esposito, Yann. *Contribution à l'inférence d'automates probabilistes*. PhD thesis, Université Aix-Marseille, 2004.

Foster, Dean P., Rodu, Jordan, and Ungar, Lyle H. Spectral dimensionality reduction for HMMs. *CoRR*, abs/1203.6130, 2012.

Gillis, N. and Vavasis, S. A. Fast and robust recursive algorithmsfor separable nonnegative matrix factorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(4):698–714, April 2014. ISSN 0162-8828.

Gillis, Nicolas. The why and how of nonnegative matrix factorization. *CoRR*, abs/1401.5226, 2014.

Glaude, Hadrien, Pietquin, Olivier, and Enderli, Cyrille. Subspace identification for predictive state representation by nuclear norm minimization. In *2014 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning, ADPRL 2014, Orlando, FL, USA, December 9-12, 2014*, pp. 1–8, 2014.

Glaude, Hadrien, Enderli, Cyrille, and Pietquin, Olivier. Non-negative spectral learning for linear sequential systems. In *Neural Information Processing - 22nd International Conference, ICONIP 2015, Istanbul, Turkey, November 9-12, 2015, Proceedings, Part II*, pp. 143–151, 2015.

Gybels, Mattias, Denis, François, and Habrard, Amaury. Some improvements of the spectral learning approach for probabilistic grammatical inference. In *Proceedings of the 12th International Conference on Grammatical Inference, ICGI 2014, Kyoto, Japan, 17-19 September 2014*, pp. 64–78, 2014.

Hsu, Daniel, Kakade, Sham M, and Zhang, Tong. A spectral algorithm for learning hidden Markov models. *Journal of Computer and System Sciences*, 78(5):1460–1480, 2012.

Kearns, Michael J., Mansour, Yishay, Ron, Dana, Rubinfeld, Ronitt, Schapire, Robert E., and Sellie, Linda. On the learnability of discrete distributions. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*, pp. 273–282, 1994.

Lőtstedt, Per. Perturbation bounds for the linear least squares problem subject to linear inequality constraints. *BIT Numerical Mathematics*, 23(4):500–519, 1983. ISSN 0006-3835. doi: 10.1007/BF01933623.

MOSEK. *The MOSEK Python optimizer API manual Version 7.1*, 2015. URL http://docs.mosek.com/7.1/pythonapi/index.html.

Mossel, Elchanan and Roch, Sébastien. Learning nonsingular phylogenies and hidden markov models. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*, pp. 366–375, 2005.

Sutskever, Ilya, Martens, James, and Hinton, Geoffrey E. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pp. 1017–1024, 2011.

Sutton, Richard S and Barto, Andrew G. *Reinforcement learning: An introduction*. MIT press, 1998.

Thon, Michael and Jaeger, Herbert. Links between multiplicity automata, observable operator models and predictive state representations–a unified learning framework. *Journal of Machine Learning Research*, 16:103–147, 2015.

Verwer, Sicco, Eyraud, Rémi, and de la Higuera, Colin. Results of the pautomac probabilistic automaton learning competition. In *Proceedings of the Eleventh International Conference on Grammatical Inference, ICGI 2012, University of Maryland, College Park, USA, September 5-8, 2012*, pp. 243–248, 2012.

Wolfe, Britton, James, Michael R., and Singh, Satinder P. Learning predictive state representations in dynamical systems without reset. In *Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005), Bonn, Germany, August 7-11, 2005*, pp. 980–987, 2005.