Learning Weighted Automata

Borja Balle[1] and Mehryar Mohri[2,3]

[1] School of Computer Science, McGill University, Montréal, Canada
[2] Courant Institute of Mathematical Sciences, New York, NY
[3] Google Research, New York, NY

1 Introduction

Weighted finite automata (WFA) are finite automata whose transitions and states are augmented with some weights, elements of a semiring. A WFA induces a function over strings. The value it assigns to an input string is the semiring sum of the weights of all paths labeled with that string, where the weight of a path is obtained by taking the semiring product of the weights of its constituent transitions, as well as those of its origin and destination states.

The mathematical theory behind WFAs, that of *rational power series*, has been extensively studied in the past [26,52,40,16] and has been more recently the topic of a dedicated handbook [23]. WFAs are widely used in modern applications, perhaps most prominently in image processing and speech recognition where the terminology of *weighted automata* seems to have been first introduced and made popular [34,43,49,41,44], in several other speech processing applications such as speech synthesis [55,1], in phonological and morphological rule compilation [35,36,47], in parsing [45], bioinformatics [25,2], sequence modeling and prediction [22], formal verification [3], in optical character recognition [18], and in many other areas.

These applications, as well as a number of theoretical questions, have strongly motivated the problem of *learning WFAs*, that is that of finding a WFA closely estimating a semiring-valued target function, using for training a finite sample of strings labeled with their target values. This problem has a rich history since its simpler instances date back to the origins of computer science. We will therefore discuss only briefly some of the key results of the literature.

A special instance of this problem is that of learning (unweighted) finite automata, which coincide with WFAs defined over the Boolean semiring. A series of negative results are known for this problem when the target itself is a finite automaton and when the complexity criterion used is the size of the automaton learned. In particular, the problem of finding a consistent deterministic finite automaton (DFA) of minimum size was shown to be NP-hard by Gold [29]. This result was later extended by Angluin [4]. Pitt and Warmuth [50] further strengthened these results by showing that even an approximation within a polynomial function of the size of the smallest consistent automaton is NP-hard. Their hardness results apply also to the case where prediction is made using non-deterministic finite automata (NFA) (see also [21]). Kearns and Valiant [37] presented for the same problem hardness results of a different nature relying on cryptographic assumptions. Their results imply that no polynomial-time algorithm can learn consistent NFAs polynomial in the size of the smallest DFA from a finite sample of accepted and rejected strings if any of the generally accepted cryptographic assumptions holds, for example if RSA public key cryptosystem is secure.

These results imply the computational intractability of the general problem of *passively* learning finite automata for several learning models, including the mistake bound model of Haussler et al. [31] or the PAC-learning model of Valiant [56]. In contrast, an *active* model of learning automata was introduced by Angluin [4,5], where the learner can make membership and equivalence queries. For this model, it was shown that finite automata can be learned in time polynomial in the size of the minimal automaton and that of the longest counter-example [4] (see also [38] and [46]).

Fewer results have been reported in the literature for the general case of learning WFAs over a non-Boolean semiring. Bergadano et al. [15] extended the positive result of [4] in the scenario where membership and equivalence queries can be made, to the problem of learning WFAs defined over any field. Using the relationship between the size of a minimal weighted automaton over a field and the rank of the corresponding Hankel matrix, the learnability of many other concepts classes such as disjoint DNF can be shown [13]. In the passive setting, the problem of learning a *probabilistic* WFA using a finite sample drawn according to the same distribution has been the subject of a series of publications in recent years using a *spectral method*, starting with the work of Hsu et al. [32] for learning hidden Markov models (HMMs). The main technique used in these publications consists of a singular value decomposition (SVD) of a Hankel matrix. Balle and Mohri [11] further showed that spectral methods combined with a constrained matrix completion algorithm can be used to learn arbitrary WFAs (not necessarily probabilistic) from finite samples drawn according to a distribution unrelated to the target WFA.

This paper surveys a number of key theoretical results and algorithms for learning WFAs. In Section 2, we introduce the main definitions and notation used throughout the paper. The notion of Hankel matrix turns out to play a key role in the definition of several learning algorithms for WFAs. In Section 3, we discuss several important properties of Hankel matrices and their use in the reconstruction of WFAs. In Section 4, we use these results to describe three algorithms for learning WFAs, as well as their theoretical guarantees.

2 Definitions and Properties

In this section, we briefly introduce some basic notions and notation related to semirings and weighted automata needed for the discussion in the following sections.

2.1 Semirings

A weighted finite automaton (WFA) $\mathcal{A}$ is a finite automaton whose transitions and states carry some weights. For various operations to be well defined, the weights must belong to a *semiring*, that is a ring that may lack negation. More formally, $(\mathbb{S}, \oplus, \otimes, \overline{0}, \overline{1})$ is a semiring if $(\mathbb{S}, \oplus, \overline{0})$ is a commutative monoid with identity element $\overline{0}$, $(\mathbb{S}, \otimes, \overline{1})$ is a monoid with identity element $\overline{1}$, $\otimes$ distributes over $\oplus$, and $\overline{0}$ is an annihilator for $\otimes$, that is $a \otimes \overline{0} = \overline{0} \otimes a = \overline{0}$ for all $a \in \mathbb{S}$.

As an example, $(\mathbb{R}_+ \cup \{+\infty\}, +, \times, 0, 1)$ is a semiring called the *probability semiring*. The semiring isomorphic to the probability semiring via the negative $\log$ is the system $(\mathbb{R} \cup \{-\infty, +\infty\}, \oplus_{\log}, +, +\infty, 0)$, where $\oplus_{\log}$ is defined by $x \oplus_{\log} y = -\log(e^{-x} + e^{-y})$; it is called the $\log$ *semiring*. The semiring derived from the $\log$ semiring via the Viterbi approximation is the system $(\mathbb{R} \cup \{-\infty, +\infty\}, \min, +, +\infty, 0)$ and is called the *tropical semiring*. It is the familiar semiring of shortest-paths algorithms.

A semiring is said to be *commutative* when the multiplicative operation $\otimes$ is commutative. It is said to be *idempotent* if $x \oplus x = x$ for all $x \in \mathbb{S}$. The Boolean semiring and the tropical semiring are idempotent.

2.2 Weighted Automata

Given an alphabet Σ, we will denote by $|x|$ the length of a string $x \in \Sigma^*$ and by ϵ the *empty string* for which $|\epsilon| = 0$.

The second operation of a semiring is used to compute the weight of a path by taking the $\otimes$-product of the weights of its constituent transitions. The first operation is used to compute the weight of any string x, by taking the $\oplus$-sum of the weights of all paths labeled with x.

For a WFA $\mathcal{A}$ defined over a semiring $(\mathbb{S}, \oplus, \otimes, \overline{0}, \overline{1})$, we denote by $Q_{\mathcal{A}}$ its finite set of states and by $E_{\mathcal{A}}$ its finite set of transitions, which are elements of $Q_{\mathcal{A}} \times \Sigma \times \mathbb{S} \times Q_{\mathcal{A}}$.[4] We will also denote by $\boldsymbol{\alpha}_{\mathcal{A}} \in \mathbb{S}^{Q_{\mathcal{A}}}$ the vector of initial weights, by $\boldsymbol{\beta}_{\mathcal{A}} \in \mathbb{S}^{Q_{\mathcal{A}}}$ the vector of final weights, and by $w_{\mathcal{A}}[e] \in \mathbb{S}$ the weight of a transition $e \in E_{\mathcal{A}}$. More generally, we denote by $w_{\mathcal{A}}[\pi]$ the weight of a path $\pi = e_1 \cdots e_n$ of $\mathcal{A}$ which is defined by the $\otimes$-product of the transitions weights: $w_{\mathcal{A}}[\pi] = w_{\mathcal{A}}[e_1] \otimes \cdots \otimes w_{\mathcal{A}}[e_n]$. For any path π, we also denote by $\mathrm{orig}[\pi]$ its origin state and by $\mathrm{dest}[\pi]$ its destination state.

It is sometimes convenient to define the set of *initial states* $I_{\mathcal{A}} = \{q \in Q_{\mathcal{A}} \colon \boldsymbol{\alpha}_{\mathcal{A}}[q] \neq \overline{0}\}$ and similarly the set of *final states* $F_{\mathcal{A}} = \{q \in Q_{\mathcal{A}} \colon \boldsymbol{\beta}_{\mathcal{A}}[q] \neq \overline{0}\}$. A path from $I_{\mathcal{A}}$ to $F_{\mathcal{A}}$ is then said to be an *accepting path*.

A WFA $\mathcal{A}$ over an alphabet Σ defines a function mapping the set of strings Σ^* to $\mathbb{S}$ that is abusively also denoted by $\mathcal{A}$ and defined as follows:

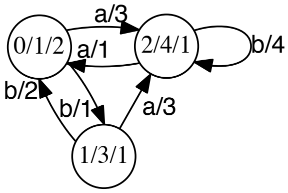
$$\forall x \in \Sigma^*, \quad \mathcal{A}(x) = \bigoplus_{\pi \in P_{\mathcal{A}}(x)} \Big(\boldsymbol{\alpha}_{\mathcal{A}}[\mathrm{orig}[\pi]] \otimes w_{\mathcal{A}}[\pi] \otimes \boldsymbol{\beta}_{\mathcal{A}}[\mathrm{dest}[\pi]] \Big),$$

where $P_{\mathcal{A}}(x)$ denotes the (finite) set of paths in $\mathcal{A}$ labeled with x. By convention, $\mathcal{A}(x) = \overline{0}$ when $P(x) = \emptyset$.

For any $a \in \Sigma$, let $\boldsymbol{\mathcal{A}}_a \in \mathbb{S}^{Q_{\mathcal{A}} \times Q_{\mathcal{A}}}$ be the matrix $[\boldsymbol{\mathcal{A}}_a]_{pq} = \oplus_{e \in P_{\mathcal{A}}(p,a,q)} w_{\mathcal{A}}[e]$, where $P_{\mathcal{A}}(p, a, q)$ is the set of transitions labeled with a from p to q. Then, (2.2) can be equivalently written as follows in terms of matrices with entries in $\mathbb{S}$:

$$\forall x = x_1 \cdots x_k \in \Sigma^*, \quad \mathcal{A}(x) = \boldsymbol{\alpha}_{\mathcal{A}}^{\top} \boldsymbol{\mathcal{A}}_{x_1} \cdots \boldsymbol{\mathcal{A}}_{x_k} \boldsymbol{\beta}_{\mathcal{A}}.$$

[4] All of our results can be straightforwardly extended to the case where $E_{\mathcal{A}}$ is a multiset, thereby allowing multiple transitions between the same two states with the same labels and weights.



(a)

$$\boldsymbol{\alpha}_{\mathcal{A}} = \begin{bmatrix}1\\3\\4\end{bmatrix} \quad \boldsymbol{\mathcal{A}}_a = \begin{bmatrix}0&0&3\\0&0&3\\1&0&0\end{bmatrix}$$

$$\boldsymbol{\beta}_{\mathcal{A}} = \begin{bmatrix}2\\1\\1\end{bmatrix} \quad \boldsymbol{\mathcal{A}}_b = \begin{bmatrix}0&1&0\\2&0&0\\0&0&4\end{bmatrix}$$

(b)

Fig. 1. (a) Example of WFA $\mathcal{A}$. Within each circle, the first number indicates the state number, the second after the slash separator the initial weight and the third the final weight. In particular, $\mathcal{A}(ab) = 1 \times 3 \times 4 \times 1 + 3 \times 3 \times 4 \times 1 + 4 \times 1 \times 1 \times 1$. (b) Corresponding initial vector $\boldsymbol{\alpha}_{\mathcal{A}}$, final vector $\boldsymbol{\beta}_{\mathcal{A}}$, and transition matrices $\boldsymbol{\mathcal{A}}_a$ and $\boldsymbol{\mathcal{A}}_b$.

This is similar to the linear representation of recognizable formal power series [52,40,16]. Figure 1 illustrates these definitions with a specific example of WFA. The *size of a WFA* is denoted by $|\mathcal{A}|$ and defined as the sum of the number of states and the number of transitions of $\mathcal{A}$: $|\mathcal{A}| = |Q_{\mathcal{A}}| + |E_{\mathcal{A}}|$. In the absence of any ambiguity, we will drop all $\mathcal{A}$-subscripts in the definitions just presented.

3 Hankel Matrices and WFA Reconstruction Algorithms

A key algebraic tool used in the design of the learning algorithms we will present is the notion of *Hankel matrix*. Thus, in this section, we present an extensive analysis of Hankel matrices and their properties. We will show how sufficiently informative finite sub-blocks of the Hankel matrix of a WFA can be used to reconstruct a WFA.

From here on, we will assume that the semiring $\mathbb{S}$ is in fact a *field*. This enables us to define the rank of a matrix with entries in $\mathbb{S}$ and devise effective algorithms for solving linear systems with unknowns and coefficients in $\mathbb{S}$. We note, however, that some of the results stated in this section can be extended to rings.

3.1 Definitions

Let $\mathbf{H} \in \mathbb{S}^{\Sigma^* \times \Sigma^*}$ be an infinite matrix with rows and columns indexed by strings in Σ^*. We denote by $\mathbf{H}(u, v)$ its entry with row index $u \in \Sigma^*$ and column index $v \in \Sigma^*$. The following definitions are essential for the rest of the paper.

Definition 1 (Hankel matrix). *We will say that* $\mathbf{H}$ *is a* Hankel matrix *if* $\mathbf{H}(u, v) = \mathbf{H}(u', v')$ *for all* $u, u, v, v' \in \Sigma^*$ *such that* $uv = u'v'$ *and will denote by* $\operatorname{rank}(\mathbf{H})$ *the rank of* $\mathbf{H}$.

Definition 2 (Hankel matrix of a function). *The* Hankel matrix $\mathbf{H}_f$ of a function $f\colon \Sigma^* \to \mathbb{S}$ *(or* formal series over $\mathbb{S}$*) is the matrix defined by* $\mathbf{H}_f(u, v) = f(uv)$*, for all* $u, v \in \Sigma^*$. *Conversely, any Hankel matrix* $\mathbf{H}$ *defines a function* $f\colon \Sigma^* \to \mathbb{S}$ *by setting* $f(u) = \mathbf{H}(u, \epsilon)$ *for all* $u \in \Sigma^*$ *and thus* $\mathbf{H} = \mathbf{H}_f$.

3.2 Hankel Matrices of Rational Functions

A function $f\colon \Sigma^* \to \mathbb{S}$ is said to be *rational* when it can be represented by a WFA $\mathcal{A}$, that is when $f(x) = \mathcal{A}(x)$ for all $x \in \Sigma^*$ [52,40,16]. The following theorem of Fliess [28] (see also [20]) provides an important characterization of rational functions in terms of the finiteness of $\mathrm{rank}(\mathbf{H}_f)$.

Theorem 1 (Fliess [28]). *Let $\mathbb{S}$ be a field. Then, the rank of the Hankel matrix $\mathbf{H}_f$ associated to a function $f\colon \Sigma^* \to \mathbb{S}$ is finite if and only if f is rational. In that case, there exists a WFA $\mathcal{A}$ representing f with $\mathrm{rank}(\mathbf{H}_f)$ states and no WFA representing f admits fewer states.*

Thus, when $\mathrm{rank}(\mathbf{H}_f) < +\infty$, a WFA representing f with $\mathrm{rank}(\mathbf{H}_f)$ states ($|Q_\mathcal{A}| = \mathrm{rank}(\mathbf{H}_\mathcal{A})$) is *minimal*. Note that this minimality is defined only in terms of the number of states, unlike the notion of minimal deterministic WFA [41,42]. In fact, such minimal WFAs often have a large number of transitions.

Proof. Suppose first that there exists a WFA $\mathcal{A}$ representing f. Then, for any $u, v \in \Sigma^*$, we can write

$$f(uv) = \mathcal{A}(uv) = (\boldsymbol{\alpha}_\mathcal{A}^\top \boldsymbol{\mathcal{A}}_u)(\boldsymbol{\mathcal{A}}_v \boldsymbol{\beta}_\mathcal{A}) \ . \tag{1}$$

Observe that $\boldsymbol{\alpha}_\mathcal{A}^\top \boldsymbol{\mathcal{A}}_u$ is a row vector in $\mathbb{S}^{1\times Q_\mathcal{A}}$ and $\boldsymbol{\mathcal{A}}_v \boldsymbol{\beta}_\mathcal{A}$ a column vector in $\mathbb{S}^{Q_\mathcal{A}\times 1}$. Let $\mathbf{P}$ be the matrix in $\mathbb{S}^{\Sigma^*\times Q_\mathcal{A}}$ defined by $\mathbf{P}_\mathcal{A}(u,\cdot) = \boldsymbol{\alpha}_\mathcal{A}^\top \boldsymbol{\mathcal{A}}_u$ for all $u \in \Sigma^*$ and $\mathbf{S}_\mathcal{A} \in \mathbb{S}^{\Sigma^*\times Q_\mathcal{A}}$ the matrix defined by $\mathbf{S}_\mathcal{A}(v,\cdot) = (\boldsymbol{\mathcal{A}}_v \boldsymbol{\beta}_\mathcal{A})^\top$ for all $v \in \Sigma^*$. Then, in view of (1), for all $u, v \in \Sigma^*$,

$$f(uv) = (\boldsymbol{\alpha}_\mathcal{A}^\top \boldsymbol{\mathcal{A}}_u)(\boldsymbol{\mathcal{A}}_v \boldsymbol{\beta}_\mathcal{A}) = (\mathbf{P}_\mathcal{A} \mathbf{S}_\mathcal{A}^\top)(u, v) \ .$$

This proves that $\mathbf{H}_f = \mathbf{P}_\mathcal{A} \mathbf{S}_\mathcal{A}^\top$. Since $\mathbf{P}_\mathcal{A}$ and $\mathbf{S}_\mathcal{A}$ are in $\mathbb{S}^{\Sigma^*\times Q_\mathcal{A}}$, the rank of $\mathbf{H}_f$ is upper bounded by $|Q_\mathcal{A}|$, the number of states of $\mathcal{A}$, and is therefore finite.

Assume now that $\mathrm{rank}(\mathbf{H}_f) = n < +\infty$. For any $v \in \Sigma^*$, we denote by $\mathbf{H}_f(\cdot, v)$ the column of $\mathbf{H}_f$ indexed by v. Let $(\mathbf{H}_f(\cdot, v_1), \ldots, \mathbf{H}_f(\cdot, v_n))$ be a basis for all columns. Then, there exist $\beta_1, \ldots, \beta_n \in \mathbb{S}$ such that the column $\mathbf{H}_f(\cdot, \epsilon)$ can be expressed as $\mathbf{H}_f(\cdot, \epsilon) = \sum_{i=1}^n \beta_i \mathbf{H}_f(\cdot, v_i)$. Since for all $w \in \Sigma^*$, $f(w) = \mathbf{H}(\epsilon, w) = \mathbf{H}(w, \epsilon) = \sum_{i=1}^n \beta_i \mathbf{H}_f(w, v_i)$, this implies that $f = \sum_{i=1}^n \beta_i \mathbf{H}_f(\cdot, v_i)$. Now, for all $i \in [1, n]$ and $a \in \Sigma$, the column $\mathbf{H}_f(\cdot, av_i)$ can also be expressed in terms of the basis: there exist (γ_{ji}^a) such that $\mathbf{H}_f(\cdot, av_i) = \sum_{j=1}^n \gamma_{ji}^a \mathbf{H}_f(\cdot, v_j)$. Let $\boldsymbol{\mathcal{A}}_a$ be the matrix defined by $(\boldsymbol{\mathcal{A}}_a)_{ji} = (\gamma_{ji}^a)$. Then, we can show by induction on the length of w that for all $w = a_1 \cdots a_k \in \Sigma^*$, $\mathbf{H}_f(\cdot, wv_i) = \sum_{j=1}^n (\boldsymbol{\mathcal{A}}_w)_{ji} \mathbf{H}_f(\cdot, v_j)$, where $\boldsymbol{\mathcal{A}}_w = \boldsymbol{\mathcal{A}}_{a_1} \cdots \boldsymbol{\mathcal{A}}_{a_k}$. Indeed, if the equality holds for w_1 and w_2, then for $w = w_1 w_2$ and for all $u \in \Sigma^*$ we have $\mathbf{H}_f(u, wv_i) = \mathbf{H}_f(uw_1, w_2 v_i)$ and:

$$\begin{aligned}
\mathbf{H}_f(uw_1, w_2 v_i) &= \sum_{j=1}^n (\boldsymbol{\mathcal{A}}_{w_2})_{ji} \mathbf{H}_f(uw_1, v_j) = \sum_{j=1}^n (\boldsymbol{\mathcal{A}}_{w_2})_{ji} \mathbf{H}_f(u, w_1 v_j) \\
&= \sum_{j=1}^n (\boldsymbol{\mathcal{A}}_{w_2})_{ji} \sum_{k=1}^n (\boldsymbol{\mathcal{A}}_{w_1})_{kj} \mathbf{H}_f(u, v_k) = \sum_{k=1}^n (\boldsymbol{\mathcal{A}}_{w_1} \boldsymbol{\mathcal{A}}_{w_2})_{ki} \mathbf{H}_f(u, v_k) \ .
\end{aligned}$$

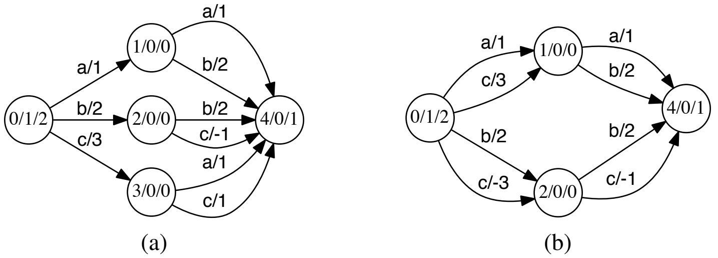


Fig. 2. Illustration of standardization. (a) WFA $\mathcal{A}$ defined over the field $(\mathbb{R}, +, \times, 0, 1)$. (b) WFA $\mathcal{B}$ obtained by standardization of $\mathcal{A}$. In this instance, the first stage of standardization leaves the WFA unchanged. In the second stage, state 3 is eliminated since it is a linear combination of states 1 and 2 in the following sense: let f_3 be the function defined by setting state 3 to be the only initial state with initial weight 1, and similarly with states 2 and 3, then, $f_3 = f_1 - f_2$.

Thus, for any $w = a_1 \cdots a_k \in \Sigma^*$,

$$f(w) = \sum_{i=1}^{n} \beta_i \mathbf{H}_f(\epsilon, wv_i) = \sum_{i=1}^{n} \beta_i \sum_{j=1}^{n} (\boldsymbol{\mathcal{A}}_w)_{ji} \mathbf{H}_f(\epsilon, v_j) = \boldsymbol{\alpha}^\top \boldsymbol{\mathcal{A}}_{a_1} \cdots \boldsymbol{\mathcal{A}}_{a_k} \boldsymbol{\beta} \ ,$$

where $\boldsymbol{\alpha}_j = \mathbf{H}_f(\epsilon, v_j)$ and $\boldsymbol{\beta}_j = \beta_j$ for all $j \in [1, n]$. This proves that f can be represented by a WFA with $n = \operatorname{rank}(\mathbf{H}_f)$ states. □

3.3 Standardization of WFAs

Theorem 1 proves the existence of a minimal WFA for the representation of a rational function f. In this section, we briefly describe an algorithm for computing a minimal WFA $\mathcal{B}$ from an input WFA $\mathcal{A}$ representing f. The first algorithm for this problem is due to Schützenberger [54] (see also [53]) and is known as a *standardization* of the representation of the linear representation of a rational power series. A more efficient version of this algorithm was later given by Cardon and Crochemore [19]. Here, we give a brief description of that algorithm.

The algorithm consists of first finding a basis $(\mathbf{v}_1, \ldots, \mathbf{v}_m)$ of row vectors in $\mathbb{S}^{1 \times Q_{\mathcal{A}}}$ for the vector space generated by $\{\boldsymbol{\alpha}_{\mathcal{A}}^\top \boldsymbol{\mathcal{A}}_w \colon w \in \Sigma^*\}$ such that for any $j \in [1, m]$ and $a \in \Sigma$, $\mathbf{v}_j \boldsymbol{\mathcal{A}}_a$ is in $\operatorname{span}(\mathbf{v}_1, \ldots, \mathbf{v}_m)$. The basis can be constructed incrementally by starting with $v_1 = \boldsymbol{\alpha}_{\mathcal{A}}^\top$ and by augmenting the current sequence of vectors $(\mathbf{v}_1, \ldots, \mathbf{v}_t)$ as follows. For any $j \in [1, t]$ and $a \in \Sigma$, the vector $\mathbf{w}$ is chosen in $\operatorname{span}(\mathbf{v}_1, \ldots, \mathbf{v}_m, \mathbf{v}_j \boldsymbol{\mathcal{A}}_a)$ such that if $\mathbf{v}_j \boldsymbol{\mathcal{A}}_a$ is linearly dependent of $(\mathbf{v}_1, \ldots, \mathbf{v}_t)$, then $\mathbf{w} = 0$; otherwise, such that $(\mathbf{v}_1, \ldots, \mathbf{v}_t, \mathbf{w})$ is triangular modulo the order of components and v_{t+1} is set to $\mathbf{w}$. Additionally, the components of $\mathbf{v}_j \boldsymbol{\mathcal{A}}_a$ are computed with respect to $(\mathbf{v}_1, \ldots, \mathbf{v}_t, \mathbf{w})$ when $\mathbf{w} \neq 0$, with respect to $(\mathbf{v}_1, \ldots, \mathbf{v}_t)$ otherwise. Testing the dependency of $\mathbf{v}_j \boldsymbol{\mathcal{A}}_a$ with respect to $(\mathbf{v}_1, \ldots, \mathbf{v}_t)$ and determining $\mathbf{w}$ such that $(\mathbf{v}_1, \ldots, \mathbf{v}_t, \mathbf{w})$ be triangular in the independent case can be done as in Gaussian elimination. This helps define a WFA $\mathcal{B}'$ equivalent to $\mathcal{A}$ and whose number of states is $\dim(\operatorname{span}(\{\boldsymbol{\alpha}_{\mathcal{A}}^\top \boldsymbol{\mathcal{A}}_w \colon w \in \Sigma^*\})) = \dim(\operatorname{span}(\{\boldsymbol{\alpha}_{\mathcal{B}'}^\top \boldsymbol{\mathcal{B}}'_w \colon w \in \Sigma^*\}))$. The components of $\mathbf{v}_j \boldsymbol{\mathcal{A}}_a$ computed by the algorithm help define the transitions of $\mathcal{B}'$. The time complexity of the algorithm is in $O(|\Sigma||Q_{\mathcal{A}}|^3)$ semiring operations since at each iteration, the complexity of determining $\mathbf{w}$ is in $O(|\Sigma||Q_{\mathcal{A}}|^2)$.

The second stage of the algorithm is symmetric. It consists of starting with $\mathcal{B}'$ and constructing a WFA $\mathcal{B}$ whose number of states is $\dim(\mathrm{span}(\{\boldsymbol{\mathcal{B}}'_w \boldsymbol{\beta}_{\mathcal{B}'} \colon w \in \Sigma^*\})) = \dim(\mathrm{span}(\{\boldsymbol{\mathcal{B}}_w \boldsymbol{\beta}_{\mathcal{B}} \colon w \in \Sigma^*\}))$. The second stage therefore coincides with the first stage if we first reverse the WFA $\mathcal{B}'$ and permute $\boldsymbol{\alpha}_{\mathcal{B}'}$ and $\boldsymbol{\beta}_{\mathcal{B}'}$. Since $|Q_{\mathcal{A}}\mathcal{B}'| \leq |Q_{\mathcal{A}}|$, the overall time complexity of the algorithm is in $O(|\Sigma||Q_{\mathcal{A}}|^3)$.

The two consecutive stages guarantee that the resulting WFA $\mathcal{B}$ is minimal.

3.4 Hankel Masks and Bases

A Hankel basis for an infinite Hankel matrix with finite rank essentially identifies a finite sub-block of that matrix which contains as much information as the infinite matrix itself. The existence of such bases is paramount for the design of learning algorithms for WFAs. Here, we will prove the existence of Hankel bases, provide bounds on their sizes, and briefly discuss the problem of finding one in practice. We start by giving several definitions.

Definition 3 (Hankel Mask). *Let $\mathcal{P}, \mathcal{S} \subseteq \Sigma^*$ be two subsets of the set of all strings. Then, the pair $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ is called a* Hankel mask. *The elements of $\mathcal{P}$ are called the* prefixes *and those of $\mathcal{S}$ the* suffixes *of the mask.*

Definition 4 (Hankel sub-block). *Let $\mathbf{H} \in \mathbb{S}^{\Sigma^* \times \Sigma^*}$ be a Hankel matrix. Given a Hankel mask $\mathcal{B} = (\mathcal{P}, \mathcal{S})$, we write $\mathbf{H}_{\mathcal{B}} \in \mathbb{S}^{\mathcal{P} \times \mathcal{S}}$ to denote the* Hankel sub-block *of $\mathbf{H}$ with rows indexed by elements of $\mathcal{P}$ and suffixes indexed by those of $\mathcal{S}$. Thus, for all $u \in \mathcal{P}$ and $v \in \mathcal{S}$ we have $\mathbf{H}_{\mathcal{B}}(u, v) = \mathbf{H}(u, v)$.*

Observe that $\mathbf{H}_{\mathcal{B}}$ inherits from $\mathbf{H}$ the Hankel property. Furthermore, since $\mathbf{H}_{\mathcal{B}}$ is a sub-block of $\mathbf{H}$, we always have $\mathrm{rank}(\mathbf{H}_{\mathcal{B}}) \leq \mathrm{rank}(\mathbf{H})$. This motivates our next definition.

Definition 5 (Hankel basis). *We say that the Hankel mask $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ is a* Hankel basis *for $\mathbf{H}$ if $\mathrm{rank}(\mathbf{H}_{\mathcal{B}}) = \mathrm{rank}(\mathbf{H})$.*

Since the rank of a matrix is upper bounded by its dimensions, all Hankel masks satisfy $\mathrm{rank}(\mathbf{H}_{\mathcal{B}}) \leq \min\{|\mathcal{P}|, |\mathcal{S}|\}$. The next result is an immediate consequence of the definition of the rank of a Hankel matrix indicating that this bound is attainable.

Proposition 1. *Let $\mathbf{H}$ be a Hankel matrix with $\mathrm{rank}(\mathbf{H}) = n$. Then there exists a Hankel basis $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ for $\mathbf{H}$ with $|\mathcal{P}| = |\mathcal{S}| = n$.*

Definition 6 (Prefix-closed and suffix-closed sets). *A subset $\mathcal{W} \subseteq \Sigma^*$ is* prefix-closed *if $w = uv \in \mathcal{W}$ implies $u \in \mathcal{W}$. Similarly, $\mathcal{W} \subseteq \Sigma^*$ is* suffix-closed *if $w = uv \in \mathcal{W}$ implies $v \in \mathcal{W}$. Note that if $\mathcal{W}$ is either prefix-closed or suffix-closed, then it must contain ϵ.*

The standardization procedure for WFA described in Section 3.3 provides further information about the structure of minimal Hankel bases.

Proposition 2. *Let $\mathbf{H}$ be a Hankel matrix with $\operatorname{rank}(\mathbf{H}) = n$. Then, there exists a Hankel basis $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ for $\mathbf{H}$ with $|\mathcal{P}| = |\mathcal{S}| = n$, where $\mathcal{P}$ is prefix-closed and $\mathcal{S}$ is suffix-closed.*

Note that, given a string $x \in \Sigma^*$, there are exactly $|x| + 1$ decompositions $x = uv$ with $u, v \in \Sigma^*$. A direct consequence of this fact is that if $\mathcal{W} \subseteq \Sigma^*$ is prefix-closed and has $|\mathcal{W}| = n$, then $|w| < n$ for every $w \in \mathcal{W}$. The same holds for suffix-closed sets. When combined with the previous proposition, this observation yields a bound on *how far* in Σ^* one needs to look in order to find a Hankel basis for a Hankel matrix $\mathbf{H}$ with rank n.

Corollary 1. *Let $\mathbf{H}$ be Hankel with $\operatorname{rank}(\mathbf{H}) = n$. Then $\mathcal{B} = (\Sigma^{<n}, \Sigma^{<n})$ is a Hankel basis for $\mathbf{H}$.*

3.5 WFA Reconstruction from Complete Minimal Masks

In this section, we describe the class of complete minimal Hankel masks, which can be used to specify the information needed to solve a WFA reconstruction problem via the Gaussian elimination algorithm in an arbitrary field. We describe the reconstruction algorithm and show that if the given mask is a Hankel basis for some Hankel matrix $\mathbf{H}_f$, then the algorithm will reconstruct a minimal WFA computing f.

Definition 7 (Hankel sub-blocks $\mathbf{H}_a$ and $\mathbf{H}_\Sigma$). *Let $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ be a Hankel mask in Σ^*. For every symbol $a \in \Sigma$, we define the Hankel mask $\mathcal{B}_a = (\mathcal{P}a, \mathcal{S})$, where $\mathcal{P}a = \{ua \colon u \in \mathcal{P}\}$. Given a Hankel matrix $\mathbf{H}$, we will use the shorthand $\mathbf{H}_a = \mathbf{H}_{\mathcal{B}_a} \in \mathbb{S}^{\mathcal{P} \times \mathcal{S}}$. Note the entries of $\mathbf{H}_a$ satisfy $\mathbf{H}_a(u, v) = \mathbf{H}(ua, v)$ for every $u \in \mathcal{P}$ and $v \in \mathcal{S}$. We denote by $\mathbf{H}_\Sigma$ the block-matrix obtained by stacking together the matrices $\mathbf{H}_a$ for all $a \in \Sigma$, that is $\mathbf{H}_\Sigma^\top = [\mathbf{H}_{a_1}^\top \cdots \mathbf{H}_{a_r}^\top]$ if $\Sigma = \{a_1, \dots, a_r\}$.*

Definition 8 (Complete and minimal Hankel masks). *A Hankel mask $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ is said to be* complete *for a Hankel matrix $\mathbf{H}$ if $\epsilon \in \mathcal{P} \cap \mathcal{S}$ and $\operatorname{rank}([\mathbf{H}_{\mathcal{B}}^\top \mid \mathbf{H}_\Sigma^\top]) = \operatorname{rank}(\mathbf{H}_{\mathcal{B}}^\top)$. A complete Hankel mask $\mathcal{B}$ is* minimal *if $\operatorname{rank}(\mathbf{H}_{\mathcal{B}}) = |\mathcal{P}|$. Note this last condition implies $|\mathcal{P}| \leq |\mathcal{S}|$.*

We now proceed to describe a WFA reconstruction algorithm that takes as input a complete minimal Hankel mask $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ for a Hankel matrix $\mathbf{H}$ and the corresponding Hankel sub-blocks $\mathbf{H}_{\mathcal{B}}$ and $\mathbf{H}_\Sigma$, and returns a WFA $\mathcal{A}$ with $k = |\mathcal{P}|$ states. Let us write $\mathcal{P} = \{u_1, \dots, u_k\}$ and $\mathcal{S} = \{v_1, \dots, v_{k'}\}$ with $u_1 = v_1 = \epsilon$. First, let $\boldsymbol{\alpha}_{\mathcal{A}}^\top = [\bar{1}, \bar{0}, \dots, \bar{0}] \in \mathbb{S}^k$ and $\boldsymbol{\beta}_{\mathcal{A}}^\top = [\mathbf{H}_{\mathcal{B}}(u_1, \epsilon), \dots, \mathbf{H}_{\mathcal{B}}(u_k, \epsilon)] = (\mathbf{H}_{\mathcal{B}}(\cdot, \epsilon))^\top$. Second, note that since $\mathcal{B}$ is complete and minimal we have $\operatorname{rank}([\mathbf{H}_{\mathcal{B}}^\top \mid \mathbf{H}_a^\top]) = \operatorname{rank}(\mathbf{H}_{\mathcal{B}}^\top) = k$ for every $a \in \Sigma$. Thus, by the Rouché–Capelli theorem, for every $a \in \Sigma$ there exists a unique $\boldsymbol{\mathcal{A}}_a \in \mathbb{S}^{k \times k}$ such that $\boldsymbol{\mathcal{A}}_a \mathbf{H}_{\mathcal{B}} = \mathbf{H}_a$. Using the Gaussian elimination algorithm, each of these systems of equations can be solved in $O(k^2(k' + k))$ arithmetic operations in $\mathbb{S}$. Thus, the arithmetic complexity of reconstructing a WFA $\mathcal{A}$ with $|\mathcal{P}|$ states from a complete minimal basis $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ is in $O(|\Sigma||\mathcal{P}|^2|\mathcal{S}|)$ [30]. If, in addition to being complete and minimal, the mask $\mathcal{B}$ is a Hankel basis for $\mathbf{H}_f$, the above procedure recovers a minimal WFA computing f.

Theorem 2. *If $\mathcal{B}$ is a complete minimal Hankel basis for $\mathbf{H}_f$, then the reconstructed WFA $\mathcal{A}$ computes f and is minimal.*

Proof. Let $\mathcal{A}'$ be a minimal WFA computing f. Recall that $\mathcal{A}'$ induces a rank factorization $\mathbf{H}_f = \mathbf{P}_{\mathcal{A}'}\mathbf{S}_{\mathcal{A}'}^\top$, which, when restricted to the Hankel basis $\mathcal{B}$ yields a rank factorization $\mathbf{H}_\mathcal{B} = \mathbf{P}'\mathbf{S}'^\top$ and associated factorizations $\mathbf{H}_a = \mathbf{P}'\boldsymbol{\mathcal{A}}_a'\mathbf{S}'^\top$ for all $a \in \Sigma$. From these, using the fact that the transition weights of $\mathcal{A}$ satisfy $\boldsymbol{\mathcal{A}}_a\mathbf{H}_\mathcal{B} = \mathbf{H}_a$ we get $\boldsymbol{\mathcal{A}}_a\mathbf{P}'\mathbf{S}'^\top = \mathbf{P}'\boldsymbol{\mathcal{A}}_a'\mathbf{S}'^\top$. Since $\mathbf{P}'$ is invertible and $\mathbf{S}'$ has full column rank, this last equation implies $\boldsymbol{\mathcal{A}}_a = \mathbf{P}'\boldsymbol{\mathcal{A}}_a'\mathbf{P}'^{-1}$. A similar argument with the initial and final weights shows that $\boldsymbol{\alpha} = \mathbf{P}'^{-1}\boldsymbol{\alpha}$ and $\boldsymbol{\beta} = \mathbf{P}'\boldsymbol{\beta}$. Therefore, we see that $\mathcal{A}$ and $\mathcal{A}'$ compute the same function, and in particular $\mathcal{A}$ computes f. Minimality is immediate by observing that $\mathcal{A}$ has $|Q_\mathcal{A}| = \operatorname{rank}(\mathbf{H}_\mathcal{B}) = \operatorname{rank}(\mathbf{H}_f)$ states. □

If the Hankel mask $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ is complete and minimal but not necessarily a Hankel basis, then the function computed by $\mathcal{A}$ will not agree with f everywhere. However, the next result shows that if $\mathcal{P}$ is prefix-free and $\mathcal{S}$ is suffix-free, then $\mathcal{A}$ will agree with f on all strings in $\mathcal{P}(\{\epsilon\} \cup \Sigma)\mathcal{S}$.

Theorem 3. *Let $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ be a complete minimal Hankel mask for $\mathbf{H}_f$. Suppose that $\mathcal{P}$ is prefix-closed and $\mathcal{S}$ is suffix-closed. Then, the WFA $\mathcal{A}$ reconstructed from $\mathbf{H}_\mathcal{B}$ and $\mathbf{H}_\Sigma$ satisfies $f(uv) = \mathcal{A}(uv)$ and $f(uav) = \mathcal{A}(uav)$ for every $u \in \mathcal{P}$, $v \in \mathcal{S}$, and $a \in \Sigma$.*

Proof. Let $k = \operatorname{rank}(\mathbf{H}_\mathcal{B}) = |\mathcal{P}|$ and $\mathcal{P} = \{u_1, \ldots, u_k\}$ with $u_1 = \epsilon$ and $|u_i| \leq |u_{i+1}|$ for all i. Let $\mathbf{H}_\mathcal{A} = \mathbf{P}_\mathcal{A}\mathbf{S}_\mathcal{A}^\top$ be the factorization induced by $\mathcal{A}$. Let us write $\mathbf{P}_\mathcal{P} \in \mathbb{S}^{\mathcal{P}\times k}$ for the sub-block of $\mathbf{P}_\mathcal{A}$ containing the rows indexed by prefixes in $\mathcal{P}$. We claim that $\mathbf{P}_\mathcal{P} = \mathbf{I}$ is the identity matrix. To see this, we will show that for $1 \leq i \leq k$ we have $\mathbf{P}_\mathcal{P}(u_i, \cdot) = \mathbf{e}_i^\top$, where $\mathbf{e}_i$ is the ith indicator vector.

By construction of $\mathcal{A}$, the case $i = 1$ holds since $\mathbf{P}_\mathcal{P}(u_i, \cdot) = \mathbf{P}_\mathcal{A}(\epsilon, \cdot) = \boldsymbol{\alpha}^\top = \mathbf{e}_1^\top$. Now, suppose the claim is true for all $1 \leq j \leq i$. Since $|u_{i+1}| \geq |u_j|$ for all $1 \leq j \leq i$ and $\mathcal{P}$ is prefix-closed, we must have $u_{i+1} = u_j a$ for some $a \in \Sigma$ and $1 \leq j \leq i$. Therefore, we have $\mathbf{P}_\mathcal{P}(u_{i+1}, \cdot) = \mathbf{P}_\mathcal{A}(u_j a, \cdot) = \mathbf{P}_\mathcal{A}(u_j, \cdot)\boldsymbol{\mathcal{A}}_a = \mathbf{e}_j^\top\boldsymbol{\mathcal{A}}_a = \boldsymbol{\mathcal{A}}_a(j, \cdot)$. Finally, we observe that because $\boldsymbol{\mathcal{A}}_a\mathbf{H}_\mathcal{B} = \mathbf{H}_a$ and $\mathbf{H}_a(u_j, \cdot) = \mathbf{H}_\mathcal{B}(u_j a, \cdot) = \mathbf{H}_\mathcal{B}(u_{i+1}, \cdot)$, when solving the system of equations for $\boldsymbol{\mathcal{A}}_a$ we will obtain $\boldsymbol{\mathcal{A}}_a(j, \cdot) = \mathbf{e}_{i+1}^\top$.

Now, let $\mathbf{S}_\mathcal{S} \in \mathbb{S}^{\mathcal{S}\times k}$ denote the sub-block of $\mathbf{S}_\mathcal{A}$ corresponding to the suffixes in $\mathcal{S}$. By the previous claim, to show that $\mathcal{A}(uv) = f(uv)$ for all $u \in \mathcal{P}$ and all $v \in \mathcal{S}$ it suffices to show that $\mathbf{S}_\mathcal{S}^\top = \mathbf{H}_\mathcal{B}$. Let $k' = |\mathcal{S}|$ and assume without loss of generality that $\mathcal{S} = \{v_1, \ldots, v_{k'}\}$ with $v_1 = \epsilon$ and $|v_{i+1}| \geq |v_i|$ for all i. Then for $i = 1$ we immediately have $\mathbf{S}_\mathcal{S}(v_1, \cdot) = \mathbf{S}_\mathcal{A}(\epsilon, \cdot) = \boldsymbol{\beta}^\top = \mathbf{H}_\mathcal{B}(\cdot, \epsilon)^\top = \mathbf{H}_\mathcal{B}(\cdot, v_1)^\top$ by the way $\boldsymbol{\beta}$ is constructed. Now, suppose we have $\mathbf{S}_\mathcal{S}(v_j, \cdot) = \mathbf{H}_\mathcal{B}(\cdot, v_j)^\top$ for all $1 \leq j \leq i$. Note we must have $v_{i+1} = a v_j$ for some $a \in \Sigma$ and some $1 \leq j \leq i$. Thus, we see that $\mathbf{S}_\mathcal{S}(v_{i+1}, \cdot) = \mathbf{S}_\mathcal{A}(a v_j, \cdot) = \mathbf{S}_\mathcal{A}(v_j, \cdot)\boldsymbol{\mathcal{A}}_a^\top = \mathbf{H}_\mathcal{B}(\cdot, v_j)^\top\boldsymbol{\mathcal{A}}_a^\top = \mathbf{H}_a(\cdot, v_j)^\top = \mathbf{H}_\mathcal{B}(\cdot, a v_j)^\top = \mathbf{H}_\mathcal{B}(\cdot, v_{i+1})^\top$.

To complete the proof it just remains to show that $\mathcal{A}(uav) = f(uav) = \mathbf{H}_a(u, v)$ for all $u \in \mathcal{P}$, $v \in \mathcal{S}$, and $a \in \Sigma$. This follows from the previous claims by noting that $\mathcal{A}(u_i a v) = \mathbf{P}_\mathcal{P}(u_i, \cdot)\boldsymbol{\mathcal{A}}_a\mathbf{S}_\mathcal{S}(v, \cdot)^\top = \mathbf{e}_i^\top\boldsymbol{\mathcal{A}}_a\mathbf{H}_\mathcal{B}(\cdot, v) = \mathbf{e}_i^\top\mathbf{H}_a(\cdot, v) = \mathbf{H}_a(u_i, v)$. □

3.6 WFA Reconstruction via Rank Factorizations

In this section, we show how a rank factorization of $\mathbf{H}_{\mathcal{B}}$ for a non-minimal complete Hankel mask $\mathcal{B}$ can be used to reconstruct a WFA. The main difference with the procedure presented in the previous sections is that here the number of states of the resulting WFA is not tied to the number of prefixes $|\mathcal{P}|$ in the mask, but to the rank of $\mathbf{H}_{\mathcal{B}}$, which can be small, even if $|\mathcal{P}|$ is large.

Let $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ be a Hankel mask in Σ^* with $\epsilon \in \mathcal{P} \cap \mathcal{S}$. Given a Hankel matrix $\mathbf{H}$, in addition to the matrices $\mathbf{H}_a = \mathbf{H}_{\mathcal{B}_a} \in \mathbb{S}^{\mathcal{P}a \times \mathcal{S}}$ for $a \in \Sigma$ introduced in the previous section, we define vectors $\mathbf{h}_{\mathcal{P}} \in \mathbb{S}^{\mathcal{P}}$ and $\mathbf{h}_{\mathcal{S}} \in \mathbb{S}^{\mathcal{S}}$ with entries given by $\mathbf{h}_{\mathcal{P}}(u) = \mathbf{H}(u, \epsilon)$ and $\mathbf{h}_{\mathcal{S}}(v) = \mathbf{H}(\epsilon, v)$. Note that the condition $\epsilon \in \mathcal{P} \cap \mathcal{S}$ implies that $\mathbf{h}_{\mathcal{P}}$ (resp. $\mathbf{h}_{\mathcal{S}}^{\top}$) can be found as a column (resp. a row) in $\mathbf{H}_{\mathcal{B}}$.

Suppose $\mathcal{B}$ is a complete Hankel mask and let k be the rank of $\mathbf{H}_{\mathcal{B}}$, $\operatorname{rank}(\mathbf{H}_{\mathcal{B}}) = k$. Then, $\mathbf{H}_{\mathcal{B}}$ admits a rank factorization $\mathbf{H}_{\mathcal{B}} = \mathbf{P}_{\mathcal{B}}\mathbf{S}_{\mathcal{B}}^{\top}$ with $\mathbf{P}_{\mathcal{B}} \in \mathbb{S}^{\mathcal{P} \times k}$ and $\mathbf{S}_{\mathcal{B}} \in \mathbb{S}^{\mathcal{S} \times k}$. Such a rank factorization can be obtained using a Gaussian elimination algorithm [30]. Next, we show how to use this rank factorization in order to reconstruct a WFA $\mathcal{A}$ with $k = |Q_{\mathcal{A}}|$ states.

The algorithm proceeds by solving a series of systems of linear equations. For the initial and final weights we find the unique solutions to $\mathbf{S}_{\mathcal{B}}\boldsymbol{\alpha}_{\mathcal{A}} = \mathbf{h}_{\mathcal{P}}$ and $\mathbf{P}_{\mathcal{B}}\boldsymbol{\beta}_{\mathcal{A}} = \mathbf{h}_{\mathcal{S}}$. Note that $\boldsymbol{\alpha}_{\mathcal{A}}$ exists and is unique since $\mathbf{S}_{\mathcal{B}}$ contains a basis of linearly independent vectors for the column-span of $\mathbf{H}_{\mathcal{B}}$ and $\mathbf{h}_{\mathcal{P}}$ is a column of $\mathbf{H}_{\mathcal{B}}$. A similar argument holds for $\boldsymbol{\beta}_{\mathcal{A}}$. For the transition weights associated with a symbol $a \in \Sigma$, we use the unique solution to the system of linear equations $\mathbf{H}_a = \mathbf{P}_{\mathcal{B}}\boldsymbol{\mathcal{A}}_a\mathbf{S}_{\mathcal{B}}^{\top}$.

One way to solve this last system of equations — and to see that indeed it admits a unique solution — is to recall that the equation for $\boldsymbol{\mathcal{A}}_a$ is equivalent to $\operatorname{vec}(\mathbf{H}_a) = (\mathbf{S}_{\mathcal{B}} \otimes_{\mathrm{K}} \mathbf{P}_{\mathcal{B}}) \operatorname{vec}(\boldsymbol{\mathcal{A}}_a)$, where $\otimes_{\mathrm{K}}$ denotes the Kronecker product between matrices, and $\operatorname{vec}(\mathbf{M})$ the result of stacking the columns of $\mathbf{M}$ into a single vector. Observe that the new system of equations admits k^2 unknowns. Its coefficients satisfy $\operatorname{rank}(\mathbf{S}_{\mathcal{B}} \otimes_{\mathrm{K}} \mathbf{P}_{\mathcal{B}}) = \operatorname{rank}(\mathbf{S}_{\mathcal{B}}) \operatorname{rank}(\mathbf{P}_{\mathcal{B}}) = k^2$ by a basic property of Kronecker products, and $\operatorname{rank}([\mathbf{S}_{\mathcal{B}} \otimes_{\mathrm{K}} \mathbf{P}_{\mathcal{B}} | \operatorname{vec}(\mathbf{H}_a)]) = \operatorname{rank}(\mathbf{S}_{\mathcal{B}} \otimes_{\mathrm{K}} \mathbf{P}_{\mathcal{B}})$ since the columns of $\mathbf{H}_a$ are linear combinations of the columns of $\mathbf{P}_{\mathcal{B}}$ because the mask $\mathcal{B}$ is complete. Thus, by the Rouché–Capelli theorem, there exists a unique solution for $\boldsymbol{\mathcal{A}}_a$. Furthermore, the solution can be found using Gaussian elimination in $O(|\mathcal{P}||\mathcal{S}|k^2)$ arithmetic operations.

Overall, the cost of reconstructing the WFA $\mathcal{A}$ starting from a complete Hankel mask takes $O(|\Sigma||\mathcal{P}||\mathcal{S}|k^2)$ arithmetic operations. As in the previous section, if, in addition to being complete, the mask $\mathcal{B}$ is a Hankel basis for some Hankel matrix $\mathbf{H}_f$, then the WFA recovered is a minimal automaton for f. The proof of this result is almost identical to that of Theorem 2 and is omitted.

Theorem 4. *If $\mathcal{B}$ is a complete Hankel basis for $\mathbf{H}_f$, then the reconstructed WFA $\mathcal{A}$ computes f and is minimal.*

3.7 WFA Reconstruction from Noisy Hankel Matrices

In the two WFA reconstruction algorithms described in the previous sections, we assumed that the Hankel sub-blocks used in the reconstruction procedure are known exactly. However, that assumption is not realistic in practice, especially when we are concerned with learning problems. We now describe a variation of the WFA reconstruction algorithm from rank factorizations that works in situations where the only available information are approximations to the Hankel sub-blocks specified by a Hankel mask.

This procedure relies in a crucial manner on the computation of a singular value decomposition (SVD), which is only possible for the real case $\mathbb{S} = \mathbb{R}$, the complex case $\mathbb{S} = \mathbb{C}$, and, in general, in the case where $\mathbb{S}$ is a field obtained as the intersection of real closed fields [48]. Since $\mathbb{S} = \mathbb{R}$ is the case which occurs more frequently in applications, and is also a case for which efficient SVD algorithms are widely available, we will present the algorithm in this section only for this case. The ideas can be straightforwardly generalized to other fields admitting an SVD.

As before, we will assume that the algorithm is given as input an arbitrary Hankel mask $\mathcal{B} = (\mathcal{P}, \mathcal{S})$. The difference is that here, instead of the exact versions of the matrices and vectors $\mathbf{H}_\mathcal{B}$, $\mathbf{H}_a$, $\mathbf{h}_\mathcal{P}$, and $\mathbf{h}_\mathcal{S}$ that represent sub-blocks of some Hankel matrix $\mathbf{H}$, the algorithm will only have access to approximate versions of these objects. For example, we are given a matrix $\widehat{\mathbf{H}}_\mathcal{B} \in \mathbb{R}^{\mathcal{P} \times \mathcal{S}}$ such that $\widehat{\mathbf{H}}_\mathcal{B} = \mathbf{H}_\mathcal{B} + \mathbf{E}_\mathcal{B}$, where $\mathbf{E}_\mathcal{B} \in \mathbb{R}^{\mathcal{P} \times \mathcal{S}}$ is a *noise* matrix. Likewise, where are given $\widehat{\mathbf{H}}_a = \mathbf{H}_a + \mathbf{E}_a$ for every $a \in \Sigma$, $\widehat{\mathbf{h}}_\mathcal{P} = \mathbf{h}_\mathcal{P} + \mathbf{e}_\mathcal{P}$, and $\widehat{\mathbf{h}}_\mathcal{S} = \mathbf{h}_\mathcal{S} + \mathbf{e}_\mathcal{S}$.

The important point to note here is that even if $\mathbf{H}_\mathcal{B}$ has small rank, say $\operatorname{rank}(\mathbf{H}_\mathcal{B}) = k \leq \operatorname{rank}(\mathbf{H}) = n$, its approximation $\widehat{\mathbf{H}}_\mathcal{B}$ may have a much larger rank, and thus, in this case, the straightforward rank factorization approach will yield a large WFA which does not necessarily resemble the one we would recover had we had access to the exact versions of $\mathbf{H}_\mathcal{B}$ and the other matrices. For example, if the error matrix $\mathbf{E}_\mathcal{B}$ is in generic position, or random, then $\mathbf{H}_\mathcal{B}$ will have full rank.

Thus, the question is now how to use these matrices to reconstruct a WFA with less states than $\operatorname{rank}(\widehat{\mathbf{H}}_\mathcal{B})$, and that ideally resembles the one we would obtain in the exact case if the amount of noise is small. The key to the solution consists of using an SVD and replace the rank factorization in the previous WFA reconstruction algorithm by a low rank approximation of $\widehat{\mathbf{H}}_\mathcal{B}$.

Now we proceed to describe the first steps of the algorithm. As input it receives the Hankel mask $\mathcal{B}$, the number of states k' that the output WFA must have, and the approximated Hankel sub-blocks described above. We start by computing the SVD of $\widehat{\mathbf{H}}_\mathcal{B}$ and using it to obtain the best rank k' approximation $\widehat{\mathbf{H}}_\mathcal{B} \approx \widehat{\mathbf{U}}\widehat{\mathbf{D}}\widehat{\mathbf{V}}^\top$, where $\widehat{\mathbf{D}} = \operatorname{diag}(\widehat{\mathfrak{s}}_1, \ldots, \widehat{\mathfrak{s}}_{k'})$ is a diagonal matrix containing the top k' singular values of $\widehat{\mathbf{H}}_\mathcal{B}$, and $\widehat{\mathbf{U}} \in \mathbb{R}^{\mathcal{P} \times k'}$ and $\widehat{\mathbf{V}} \in \mathbb{R}^{\mathcal{S} \times k'}$ contain the associated left and right singular vectors respectively. With this notation, one can see that now $\widehat{\mathbf{P}}_\mathcal{B} = \widehat{\mathbf{U}}\widehat{\mathbf{D}}$ and $\widehat{\mathbf{S}}_\mathcal{B} = \widehat{\mathbf{V}}$ provide a rank factorization $\widehat{\mathbf{P}}_\mathcal{B}\widehat{\mathbf{S}}_\mathcal{B}$ of the best rank k' approximation to $\widehat{\mathbf{H}}_\mathcal{B}$.

The next natural step in the algorithms would be to solve the following systems of linear equations in order to reconstruct a WFA: $\widehat{\mathbf{S}}_\mathcal{B}\widehat{\boldsymbol{\alpha}} = \widehat{\mathbf{h}}_\mathcal{P}$, $\widehat{\mathbf{P}}_\mathcal{B}\widehat{\boldsymbol{\beta}} = \widehat{\mathbf{h}}_\mathcal{S}$, and $(\widehat{\mathbf{S}}_\mathcal{B} \otimes_{\mathrm{K}} \widehat{\mathbf{P}}_\mathcal{B}) \operatorname{vec}(\widehat{\boldsymbol{\mathcal{A}}}_a) = \operatorname{vec}(\widehat{\mathbf{H}}_a)$ for every $a \in \Sigma$. There is, however, an obstruction to the direct application of this strategy in this case: these equations are no longer guaranteed to admit a unique solution. Due to the errors in the Hankel sub-blocks introduced by the approximation, these equations might now be unsatisfiable or not admit a unique solution. Thus, we will follow a least-squares approach and look for a solution to these equations that minimizes the norm of the residual. A way to express these solutions in closed-form is via the Moore–Penrose pseudo-inverse $\mathbf{M}^+ \in \mathbb{R}^{d_2 \times d_1}$ of a matrix $\mathbf{M} \in \mathbb{R}^{d_1 \times d_2}$. In particular, given a linear system of equations $\mathbf{M}\mathbf{x} = \mathbf{b}$ the pseudo-inverse yields a solution $\mathbf{x} = \mathbf{M}^+\mathbf{b}$ that satisfies the equation if it is satisfiable, and that minimizes the error $\|\mathbf{M}\mathbf{x} - \mathbf{b}\|$ otherwise.

Now we proceed to describe the rest of the algorithm, which essentially applies this strategy to solve the linear systems given above. For the initial and final weights this yields $\widehat{\boldsymbol{\alpha}} = \widehat{\mathbf{S}}_{\mathcal{B}}^+ \widehat{\mathbf{h}}_{\mathcal{P}}$ and $\widehat{\boldsymbol{\beta}} = \widehat{\mathbf{P}}_{\mathcal{B}}^+ \widehat{\mathbf{h}}_{\mathcal{S}}$. In our case, these are easy to compute because by properties of the pseudo-inverse it can be shown that $\widehat{\mathbf{P}}_{\mathcal{B}}^+ = (\widehat{\mathbf{U}}\widehat{\mathbf{D}})^+ = \widehat{\mathbf{D}}^{-1}\widehat{\mathbf{U}}^\top$ and $\widehat{\mathbf{S}}_{\mathcal{B}}^+ = \widehat{\mathbf{V}}^+ = \widehat{\mathbf{V}}^\top$. For the transition weights, a short algebraic calculation shows that $(\widehat{\mathbf{S}}_{\mathcal{B}} \otimes_{\mathrm{K}} \widehat{\mathbf{P}}_{\mathcal{B}})^+ = (\widehat{\mathbf{S}}_{\mathcal{B}}^+ \otimes_{\mathrm{K}} \widehat{\mathbf{P}}_{\mathcal{B}}^+) = (\widehat{\mathbf{V}}^\top \otimes_{\mathrm{K}} \widehat{\mathbf{D}}^{-1}\widehat{\mathbf{U}}^\top)$. Substituting into $\operatorname{vec}(\widehat{\boldsymbol{\mathcal{A}}}_a) = (\widehat{\mathbf{S}}_{\mathcal{B}} \otimes_{\mathrm{K}} \widehat{\mathbf{P}}_{\mathcal{B}})^+ \operatorname{vec}(\widehat{\mathbf{H}}_a)$ and applying the equivalence between vectorized and unvectorized systems of linear equations, we obtain the expression $\widehat{\boldsymbol{\mathcal{A}}}_a = \widehat{\mathbf{D}}^{-1}\widehat{\mathbf{U}}^\top \widehat{\mathbf{H}}_a \widehat{\mathbf{V}}$.

Overall, the complexity of this process is dominated by the low-rank SVD computation, which takes $O(|\mathcal{P}||\mathcal{S}|k')$ arithmetic operations. Hence, the arithmetic complexity of computing the WFA $\widehat{\mathcal{A}}$ with k' states given by $\widehat{\boldsymbol{\alpha}}$, $\widehat{\boldsymbol{\beta}}$, and $\widehat{\boldsymbol{\mathcal{A}}}_a$ for $a \in \Sigma$, is in $O(|\Sigma||\mathcal{P}||\mathcal{S}|k')$.

The main result of this section is a bound on the sensitivity of this algorithm to the magnitude of the noise. To make this more precise, we need two ingredients. The first is a precise way to quantify the error in the approximations. Different choices lead to slightly different results, but in order to illustrate the point we will simply use the Euclidean norm for vectors and the Frobenius norm for matrices. Thus, we will define $\varepsilon_{\mathcal{B}} = \|\mathbf{E}_{\mathcal{B}}\|_F$, $\varepsilon_a = \|\mathbf{E}_a\|_F$ for every $a \in \Sigma$, $\varepsilon_{\mathcal{P}} = \|\mathbf{e}_{\mathcal{P}}\|_2$, and $\varepsilon_{\mathcal{S}} = \|\mathbf{e}_{\mathcal{S}}\|_2$. For convenience we will also write $\varepsilon = \max\{\varepsilon_{\mathcal{B}}, \varepsilon_{a_1}, \ldots, \varepsilon_{a_r}, \varepsilon_{\mathcal{P}}, \varepsilon_{\mathcal{S}}\}$. The second ingredient is to determine what would the output of the algorithm be if the input had no noise. For that purpose, let us assume that $k' = \operatorname{rank}(\mathbf{H}_{\mathcal{B}}) = k$ and $\varepsilon = 0$. In that case, we have $\widehat{\mathbf{H}}_{\mathcal{B}} = \mathbf{H}_{\mathcal{B}}$ and the SVD of rank k yields an exact rank factorization $\mathbf{H}_{\mathcal{B}} = \mathbf{P}\mathbf{S}^\top = (\mathbf{U}\mathbf{D})(\mathbf{V})^\top$. Thus, the algorithm returns a WFA with k states given by $\boldsymbol{\alpha} = \mathbf{V}^\top \mathbf{h}_{\mathcal{P}}$, $\boldsymbol{\beta} = \mathbf{D}^{-1}\mathbf{U}^\top \mathbf{h}_{\mathcal{S}}$, and $\boldsymbol{\mathcal{A}}_a = \mathbf{D}^{-1}\mathbf{U}^\top \mathbf{H}_a \mathbf{V}$, where we dropped the hat notation to indicate that we are in the case $\varepsilon = 0$. For this automaton, a direct application of Theorem 4 yields the following result, which shows that this is essentially a generalization of the WFA reconstruction algorithm based on rank factorizations.

Corollary 2. *Suppose $k' = \operatorname{rank}(\mathbf{H}_{\mathcal{B}})$ and $\varepsilon = 0$. If $\mathcal{B}$ is a complete basis for $\mathbf{H}_f$, then the reconstructed WFA $\mathcal{A}$ computes f and is minimal.*

The most important result about the WFA reconstructions algorithm based on SVD is the following, which bounds the error between the noisy and the noiseless cases.

Theorem 5. *Suppose $k' = \operatorname{rank}(\mathbf{H}_{\mathcal{B}})$. Let $\mathcal{A}$ denote the WFA obtained in the case $\varepsilon = 0$ and $\widehat{\mathcal{A}}$ the WFA obtained in the noisy case. Then, the following approximation guarantee holds as $\varepsilon \to 0$:*

$$\Delta = \max\{\|\boldsymbol{\alpha} - \widehat{\boldsymbol{\alpha}}\|_2, \|\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}}\|_2, \|\boldsymbol{\mathcal{A}}_{a_1} - \widehat{\boldsymbol{\mathcal{A}}}_{a_1}\|_F, \ldots, \|\boldsymbol{\mathcal{A}}_{a_r} - \widehat{\boldsymbol{\mathcal{A}}}_{a_r}\|_F\} = O(\varepsilon).$$

The proof of this results is technical and goes beyond the scope of the present survey. Essentially, it involves a detailed analysis using perturbation theory for singular values and vectors (see [9, Chapter 5] for details).

4 Algorithms for Learning WFAs

In this section, we show how the reconstruction techniques described in the previous section can be used in the design of algorithms for learning WFAs. We describe three WFA learning algorithms, each designed for a different learning scenario. The scenarios mainly differ by the way the data about the target function $f \colon \Sigma^* \to \mathbb{S}$ is gathered: exact learning from membership and equivalence queries (Section 4.1), PAC learning (Probably approximately correct learning) of a probability distribution represented by a WFA from i.i.d. samples (Section 4.2), and statistical learning of WFA from general string–label pairs (Section 4.3).

We also present learning guarantees in each case, thereby showcasing an important trade-off between degree of fidelity of the information collected versus quality of the learned WFA with respect to a target automaton or distribution. Of the three scenarios, only the first one can learn WFA over an arbitrary field $\mathbb{S}$; in the other two scenarios we restrict ourselves only to the case $\mathbb{S} = \mathbb{R}$.

4.1 Learning WFAs From Queries

In this section, we describe an algorithm for learning WFAs defined over an arbitrary field $\mathbb{S}$. The algorithm was first presented in [14] for the special case $\mathbb{S} = \mathbb{Q}$ and later generalized to arbitrary fields in [15]. It can be interpreted as a direct generalization of Angluin's classical algorithm for learning DFAs from membership and equivalence queries [5] and can be further applied to other learning problems (see [12,13]).

The learning scenario for this algorithm coincides with the active learning scenario defined and adopted by Angluin [5] for learning (unweighted) automata. In this scenario, given a target rational function $f \colon \Sigma^* \to \mathbb{S}$ the learner can make the following two types of queries to which an oracle responds:

- *membership queries* MQ_f: the learner requests the target value $f(w)$ of a string $w \in \Sigma^*$ and receives that value;
- *equivalence queries* EQ_f: the learner conjectures a WFA $\mathcal{A}$; he receives the response yes if f can be computed by $\mathcal{A}$, a counter-example $w \in \Sigma^*$ with $f(w) \neq \mathcal{A}(w)$ otherwise.

The objective of the learner is to determine exactly a WFA $\mathcal{A}$ representing f. We will denote by n the unknown rank of the Hankel matrix of f, $n = \mathrm{rank}(\mathbf{H}_f)$.

The main idea behind the algorithm is to build a complete minimal Hankel basis $\mathcal{B}$ for $\mathbf{H}_f$, fill the associated Hankel sub-blocks $\mathbf{H}_{\mathcal{B}}$ and $\mathbf{H}_{\Sigma}$ by making a series of calls to MQ_f, and then reconstruct the corresponding WFA using the Gaussian elimination algorithm described in Section 3.5. In order to find such a basis $\mathcal{B}$ several intermediate complete minimal Hankel masks are considered. For each, the corresponding WFA is reconstructed using information collected from membership queries, and the counter-examples supplied by the equivalence queries used to extend the current Hankel mask.

Given two bases $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ and $\mathcal{B}' = (\mathcal{P}', \mathcal{S}')$, we will write in short $\mathcal{B} \subseteq \mathcal{B}'$ for $\mathcal{P} \subseteq \mathcal{P}'$ and $\mathcal{S} \subseteq \mathcal{S}'$. The algorithm constructs a sequence of complete minimal Hankel masks $\mathcal{B}_0 \subseteq \mathcal{B}_1 \subseteq \cdots \subseteq \mathcal{B}_d$, where the last mask $\mathcal{B}_d$ is a Hankel basis for $\mathbf{H}_f$. At each step, the inequality $\mathrm{rank}(\mathbf{H}_{\mathcal{B}_{i+1}}) > \mathrm{rank}(\mathbf{H}_{\mathcal{B}_i})$ holds, which guarantees that the total number of iterations is at most $d \leq n$. The starting mask is $\mathcal{B}_0 = (\{\epsilon\}, \{\epsilon\})$, which is clearly a complete and minimal mask.

The main inductive step is given by the following procedure. First, given $\mathcal{B}_i = (\mathcal{P}_i, \mathcal{S}_i)$ with $i \geq 0$, the algorithm reconstructs a WFA $\mathcal{A}_i$ by filling the corresponding Hankel sub-blocks using calls to MQ_f and then applying the reconstruction algorithm of Section 3.5. Second, it makes an equivalence query $\mathsf{EQ}_f(\mathcal{A}_i)$. If the answer is yes, the algorithm terminates. Otherwise, it receives a counter-example $w \in \Sigma^*$ such that $\mathcal{A}_i(w) \neq f(w)$. This is used to build the new Hankel mask $\mathcal{B}_{i+1} = (\mathcal{P}_{i+1}, \mathcal{S}_{i+1})$ as follows:

1. find a decomposition $w = uav$ where u is the longest prefix of w in $\mathcal{P}_i$;
2. let $\mathcal{S}_{i+1} = \mathcal{S}_i \cup \mathrm{suffs}(v)$, where $\mathrm{suffs}(v)$ is the set of all suffixes of v;
3. starting from $\mathcal{P}_{i+1} = \mathcal{P}_i$, and while $\mathrm{rank}([\mathbf{H}_{\mathcal{B}_{i+1}}^\top \mid \mathbf{H}_\Sigma^\top]) > s\,\mathrm{rank}(\mathbf{H}_{\mathcal{B}_{i+1}})$, keep adding to $\mathcal{P}_{i+1}$ prefixes $ua \in \mathcal{P}_{i+1}\Sigma$ such that $\mathrm{rank}([\mathbf{H}_{\mathcal{B}_{i+1}}^\top \mid \mathbf{H}_a(u,:)^\top]) = \mathrm{rank}(\mathbf{H}_{\mathcal{B}_{i+1}}) + 1$.

Note the resulting mask $\mathcal{B}_{i+1}$ is complete by construction and minimal because only prefixes that increase the rank of $\mathbf{H}_{\mathcal{B}_{i+1}}$ are added to $\mathcal{P}_{i+1}$. Also note that the algorithm maintains the property that $\mathcal{P}_i$ is prefix-closed and $\mathcal{S}_i$ suffix-closed. It is clear that if the algorithm terminates, it returns the correct answer. To prove that the algorithm terminates it suffices to show that at each iteration the inequality $|\mathcal{P}_{i+1}| > |\mathcal{P}_i|$ holds since this will guarantee that at each iteration the rank of $\mathbf{H}_{\mathcal{B}_i}$ increases. Since this rank can be at most $n = \mathrm{rank}(\mathbf{H}_f)$, and since whenever $\mathrm{rank}(\mathbf{H}_{\mathcal{B}_i}) = \mathrm{rank}(\mathbf{H}_f)$ $\mathcal{B}_i$ is a complete minimal Hankel basis, Theorem 2 then shows that the WFA $\mathcal{A}_i$ computes f. The termination of the algorithm is guaranteed by the following result.

Lemma 1. *Let $\mathcal{B}_i' = (\mathcal{P}_i, \mathcal{S}_{i+1})$, where $\mathcal{S}_{i+1}$ is the set of suffixes obtained after processing the counter-example w received from the $(i+1)$th call to EQ_f. Then, the following inequality holds:* $\mathrm{rank}([\mathbf{H}_{\mathcal{B}_i'}^\top \mid \mathbf{H}_\Sigma^\top]) > \mathrm{rank}(\mathbf{H}_{\mathcal{B}_i'}^\top)$.

Proof. Suppose that $\mathrm{rank}([\mathbf{H}_{\mathcal{B}_i'}^\top \mid \mathbf{H}_\Sigma^\top]) = \mathrm{rank}(\mathbf{H}_{\mathcal{B}_i'}^\top)$ and let $\mathcal{A}_i'$ be the WFA reconstructed from $\mathcal{B}_i'$ by the algorithm in Section 3.5. Since $\mathcal{B}_i$ and $\mathcal{B}_i'$ share the same prefixes, both are minimal and complete, and $\mathcal{S}_i \subseteq \mathcal{S}_{i+1}$, then $\mathcal{A}_i$ and $\mathcal{A}_i'$ must compute the same function. Thus, we have $f(w) \neq \mathcal{A}_i(w) = \mathcal{A}_i'(w)$. On the other hand, $w = uav$ with $u \in \mathcal{P}_i$ and $v \in \mathcal{S}_{i+1}$. Thus, in the matrix $\mathbf{H}_a$ used to reconstruct $\mathcal{A}_i'$ we have $\mathbf{H}_a(u,v) = f(w)$, and by Theorem 3 it holds that $\mathcal{A}_i'(w) = f(w)$. We conclude by contradiction that $\mathrm{rank}([\mathbf{H}_{\mathcal{B}_i'}^\top | \mathbf{H}_\Sigma^\top]) > \mathrm{rank}(\mathbf{H}_{\mathcal{B}_i'}^\top)$. □

We can now bound the number of queries made by the algorithm. First observe that the number of calls to EQ_f is $O(n)$ since one such call is made for each of the $d+1$ Hankel masks. To bound the number of calls to MQ_f, note that since we have $\mathcal{B}_i \subseteq \mathcal{B}_{i+1}$ for each i, at each stage most of the queries needed to fill $\mathbf{H}_{\mathcal{B}_{i+1}}$ have already been asked in previous iterations. Thus, it suffices to count the number of MQ_f queries needed to fill the matrices corresponding to the last Hankel mask $\mathcal{B}_d = (\mathcal{P}_d, \mathcal{S}_d)$. This number is clearly $(|\Sigma| + 1)|\mathcal{P}_d||\mathcal{S}_d|$.

Let L denote the length of the longest counter-example returned by the successive calls to EQ_f, we have $|\mathcal{S}_d| \leq 1+dL$. This, combined with $|\mathcal{P}_d| = n$, shows that the total number of calls to MQ_f is in $O(|\Sigma|n^2 L)$. Note that this complexity is not optimal: [17] give an improved technique for processing counter-examples that yields an algorithm making only $O(|\Sigma|n^2 \log(L))$ calls to MQ_f.

4.2 Learning Stochastic WFAs from I.I.D. Samples

A *stochastic WFA* is a WFA computing a probability distribution. In this section, we consider the problem of learning a stochastic WFA and therefore assume that $\mathbb{S} = \mathbb{R}$. The learning scenario commonly adopted for stochastic WFAs is one where the learner receives a finite set of strings sampled i.i.d. from the target stochastic WFA. The objective of the learner is to use this training sample to learn a WFA computing a function close the target distribution with respect to some measure of accuracy.

In this section, we present an algorithm for this problem which consists of first using the training sample to estimate a sub-block of the Hankel matrix of the target WFA, and next of using the algorithm described in Section 3.7 to reconstruct a WFA based on those estimates. Several variants of this algorithm can be found in the literature, including [33,7] for the first such algorithms based on SVD, [10] for variants using prefix and substring statistics, and [6,9] for detailed analyses and further references.

A stochastic WFA over Σ is one that computes a probability distribution over Σ^*, that is, a WFA $\mathcal{A}$ with $\mathcal{A}(w) \geq 0$ for all $w \in \Sigma^*$ and $\sum_{w \in \Sigma^*} \mathcal{A}(w) = 1$. Probabilistic automata with stopping probabilities or absorbing states are typical examples of stochastic WFAs in this class (see [57,58,24] for a discussion of the relations between different finite-state machines computing probability distributions).

Let $\mathcal{A}$ be a fixed unknown target stochastic WFA $\mathcal{A}$. We assume that the learning algorithm receives a sample $S = (w_1, \ldots, w_m) \in (\Sigma^*)^m$ of m strings sampled i.i.d. from the distribution computed by $\mathcal{A}$. In addition to S, the algorithm receives the alphabet Σ, a number of states n that the output automaton should have, and a finite Hankel mask $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ with $n \leq \min\{|\mathcal{P}|, |\mathcal{S}|\}$.

The first step of the algorithm is to compute empirical estimates of the matrices and vectors required by the SVD-based WFA reconstruction algorithm of Section 3.7: $\widehat{\mathbf{H}}_{\mathcal{B}}$, $\widehat{\mathbf{H}}_a$ for $a \in \Sigma$, $\widehat{\mathbf{h}}_{\mathcal{P}}$, and $\widehat{\mathbf{h}}_{\mathcal{S}}$. This is done by assigning to each entry in these matrices and vectors the relative frequency of the corresponding string in the sample $S = (w_1, \ldots, w_m)$. For example, for $u \in \mathcal{P}$ and $v \in \mathcal{S}$ the algorithm sets

$$\widehat{\mathbf{H}}_{\mathcal{B}}(u, v) = \frac{1}{m} \sum_{i=1}^{m} \mathbb{I}[w_i = uv] \enspace .$$

The same is done for $\widehat{\mathbf{H}}_a$, $\widehat{\mathbf{h}}_{\mathcal{P}}$, and $\widehat{\mathbf{h}}_{\mathcal{S}}$. These approximations are then used by the WFA reconstruction algorithm to obtain an automaton $\widehat{\mathcal{A}}$ with n states.

The empirical probabilities used in the estimations of the Hankel sub-blocks converge to the true probabilities as $m \to \infty$. One can also expect that the difference between the unknown probability distribution f and the function computed by $\widehat{\mathcal{A}}$ decreases as m increases. The next theorem gives a stronger guarantee which holds for finite samples, as opposed to a result holding in the limit. It is a *probably approximately correct* (PAC) learning guarantee: for a sample size m polynomial in the size of the $1/\epsilon$ where ϵ is the precision sought, $\log(1/\delta)$ where δ is the confidence parameter and several other parameters including $1/\mathfrak{s}_n(\mathbf{H}_{\mathcal{B}})$ where $\mathfrak{s}_n(\mathbf{H}_{\mathcal{B}})$ is the singular value of $\mathbf{H}_{\mathcal{B}}$ and the string length L, the WFA $\widehat{\mathcal{A}}$ returned by the algorithm is ϵ-close to f for the norm-1 over the set of strings of length at most L.

Theorem 6. *Let $\varepsilon > 0$. Then, for any $\delta \in (0,1)$, with probability at least $1-\delta$ over the draw of a sample S of size $m \geq p(|\Sigma|, n, |\mathcal{P}|, |\mathcal{S}|, 1/\mathfrak{s}_n(\mathbf{H}_{\mathcal{B}}), L, 1/\varepsilon, \log(1/\delta))$ from the (target) probability distribution f, where p is a polynomial, the WFA $\widehat{\mathcal{A}}$ returned by the algorithm after receiving S, a complete Hankel basis $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ for $\mathbf{H}_f$ and $n = \operatorname{rank}(\mathbf{H}_f)$ verifies the following inequality:*

$$\sum_{w \in \Sigma^{\leq L}} |f(w) - \widehat{\mathcal{A}}(w)| \leq \varepsilon \ .$$

The proof of this result admits three components: Theorem 5, a concentration bound for the estimates $\widehat{\mathbf{H}}_{\mathcal{B}}$, and a bound relating accuracy in transition weights between $\mathcal{A}$ and $\widehat{\mathcal{A}}$ to accuracy in the function they compute (see [33,6,9] for detailed proofs).

4.3 Learning WFAs from String–Value Pairs

In this section, we present an algorithm for learning WFAs in a more general scenario than the previous ones. This scenario was first introduced in [11]. The learning algorithm for WFAs described here is also due to [11].

Here, as in the standard supervised learning, the learner receives a labeled sample $S = ((w_1, y_1), \ldots, (w_m, y_m)) \in (\Sigma^* \times \mathbb{S})^m$ containing m string–value pairs $(w_i, y_i) \in \Sigma^* \times \mathbb{S}$, drawn i.i.d. according to some unknown distribution $\mathcal{D}$. The learning problem consists of finding a WFA $\mathcal{A}$ with small expected loss, that is with small $\mathbb{E}_{(w,y)\sim\mathcal{D}}[\ell(\mathcal{A}(w), y)]$, where ℓ is a loss function defined over semiring pairs. We will consider again here the case $\mathbb{S} = \mathbb{R}$. The problem is then an instance of a regression learning problem. The loss function $\ell \colon \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$ is used to measure the closeness of the labels. Some common choices for ℓ are the quadratic loss defined for all $y, y' \in \mathbb{R}$ by $\ell_2(y, y') = (y - y')^2$ and the absolute loss defined by $\ell_1(y, y') = |y - y'|$.

Note that in this formulation we did not assume that the labels y in pairs (x, y) drawn from $\mathcal{D}$ are computed by some WFA. Thus, in learning-theoretic terms, we consider an *agnostic setting*.

Note also that one could find a WFA $\mathcal{A}$ consistent with the labeled sample, that is such that $\mathcal{A}(w_i) = y_i$ for all $i \in [1, m]$. But, such a WFA could be large and might not benefit from a favorable expected loss. Furthermore, it was recently shown in [39] that the problem of finding the smallest WFA $\mathcal{A}$ consistent with the labeled sample is computationally hard.

A WFA minimizing the empirical loss $\frac{1}{m}\sum_{i=1}^{m} \ell(\mathcal{A}(w_i), y_i)$ could *overfit* the training sample and typically would not benefit from favorable learning guarantees unless it is selected out of a less *complex* sub-family of WFAs. The algorithm we describe here avoids overfitting by constraining the choice of a WFA in two ways: by restricting the number of states, and by controlling the norm of a certain Hankel matrix.

The algorithm works in two stages. In the first stage, the sample S is used to find a sub-block of a Hankel matrix on a given mask. The second stage uses this Hankel block to reconstruct a WFA with a given number of states using the SVD-based method from Section 3.7. The algorithm receives as input the sample S, the alphabet Σ, a Hankel mask $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ with $\epsilon \in \mathcal{P} \cap \mathcal{S}$, a number of states $k \leq \min\{|\mathcal{P}|, |\mathcal{S}|\}$, a convex loss function $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$, and a regularization parameter $\lambda > 0$.

The first stage builds a basis $\mathcal{B}' = (\mathcal{P}', \mathcal{S})$ with $\mathcal{P}' = \mathcal{P} \cup \mathcal{P}\Sigma$ and a modified sample S' containing only those $(w_i, y_i) \in S$ such that $w_i \in \mathcal{P}'\mathcal{S}$. Then, the algorithm solves the convex optimization problem

$$\widehat{\mathbf{H}}_{\mathcal{B}'} \in \operatorname*{argmin}_{\mathbf{H} \in \mathbb{H}_{\mathcal{B}'}} \frac{1}{|S'|} \sum_{(w,y) \in S'} \ell(\mathbf{H}(w), y) + \lambda \|\mathbf{H}\|_* \ ,$$

where $\mathbb{H}_{\mathcal{B}'}$ denotes the set of all Hankel matrices $\mathbf{H} \in \mathbb{R}^{\mathcal{P}' \times \mathcal{S}}$, $\mathbf{H}(w)$ denotes $\mathbf{H}(u, v)$ for some arbitrary decomposition $w = uv$ with $u \in \mathcal{P}$ and $v \in \mathcal{S}$, and where $\|\mathbf{H}\|_*$ denotes the nuclear norm of $\mathbf{H}$ defined as the sum of the singular values of $\mathbf{H}$.

The second stage of the algorithm starts by extracting from $\widehat{\mathbf{H}}_{\mathcal{B}'}$ the Hankel sub-blocks associated with the Hankel mask $\mathcal{B}$: $\widehat{\mathbf{H}}_{\mathcal{B}}$, $\widehat{\mathbf{H}}_a$ for $a \in \Sigma$, $\mathbf{h}_{\mathcal{P}}$, and $\mathbf{h}_{\mathcal{S}}$. Then, it uses the SVD-based WFA reconstruction algorithm of Section 3.7 to obtain a WFA $\widehat{\mathcal{A}}$ with k states.

The design of the algorithm, and in particular the choice of the nuclear norm as a regularization term for finding the Hankel matrix $\widehat{\mathbf{H}}_{\mathcal{B}'}$ is supported by several properties. First, the nuclear norm is a convex surrogate for the rank function commonly used in machine learning algorithms [27]. By Theorem 1, low-rank Hankel matrices correspond to WFAs with small numbers of states, thus it favors the selection of smaller WFAs by the algorithm. A second justification is given by the following theorem, which provides a guarantee for learning with WFAs in terms of the nuclear norm of the associated Hankel matrix.

Let $M > 0$ and define $\tau_M : \mathbb{R} \to \mathbb{R}$ as the function defined by $\tau_M(y) = \operatorname{sign}(y) M$ if $|y| > M$, $\tau_M(y) = y$ otherwise. Let $S = ((w_1, y_1), \ldots, (w_m, y_m)) \in (\Sigma^* \times \mathbb{R})^m$. Given a decomposition $w_i = u^i v^i$, for any $1 \leq i \leq m$, we define $U_S = \max_{u \in \Sigma^*} |\{i : u^i = u\}|$ and $V_S = \max_{v \in \Sigma^*} |\{i : v^i = v\}|$. A measure of the complexity of S that will appear in the next theorem is $W_S = \min \max\{U_S, V_S\}$, where the minimum is taken over all possible decompositions of the strings w_i in S. For any $R > 0$, let $\mathcal{F}_R$ denote the following class of functions

$$\mathcal{F}_R = \{f(w) = \tau_M(\mathcal{A}(w)) : \ \mathcal{A} \text{ WFA}, \|\mathbf{H}_f\|_* \leq R\} \ .$$

The following gives a learning bound for the algorithm just discussed.

Theorem 7. *Let ℓ_1 denote the absolute loss. Assume that there exists $M > 0$ such that $\mathbb{P}_{(w,y) \sim D}[|y| \leq M] = 1$. Then, for any $\delta > 0$, with probability at least $1 - \delta$ over the draw of an i.i.d. sample S of size m from $\mathcal{D}$, the following inequality holds* *simultaneously for all $f \in \mathcal{F}_R$:*

$$\mathop{\mathbb{E}}_{(w,y)\sim D}[\ell_1(f(w),y)] \leq \frac{1}{m}\sum_{i=1}^{m}\ell_1(f(w_i),y_i) + 3M\sqrt{\frac{\log(\frac{2}{\delta})}{2m}} + O\left(\frac{R\big(\log(m+1) + \sqrt{W_S \log(m+1)}\big)}{m}\right).$$

A similar result was first proven in [11] using a Frobenius norm regularizer instead of a nuclear norm. The analysis in [11] was based on a stability argument, and it is not clear how to extend it to the nuclear norm case, which is known to perform better than the Frobenius norm in some applications [51]. Theorem 7 is proven using a Rademacher complexity analysis of WFAs recently given by [8].

5 Conclusion

We presented a detailed survey of modern algorithms for learning WFAs. We highlighted the key role played by the notion of Hankel matrix and its properties in the design of these learning algorithms which are designed for different scenarios. These properties and the algorithms we described could inspire other variants of these algorithms as well as other algorithms.

Acknowledgments

This work was partly funded by the NSF award IIS-1117591 and NSERC.

References

1. Allauzen, C., Mohri, M., Riley, M.: Statistical modeling for unit selection in speech synthesis. In: Proceedings of ACL (2004)
2. Allauzen, C., Mohri, M., Talwalkar, A.: Sequence kernels for predicting protein essentiality. In: Proceedings of ICML (2008)
3. Aminof, B., Kupferman, O., Lampert, R.: Formal analysis of online algorithms. In: Proceedings of ATVA (2011)
4. Angluin, D.: On the complexity of minimum inference of regular sets. Information and Control 3(39) (1978)
5. Angluin, D.: Learning regular sets from queries and counterexamples. Information and computation 75(2) (1987)
6. Bailly, R.: Méthodes spectrales pour l'inférence grammaticale probabiliste de langages stochastiques rationnels. Ph.D. thesis, Aix-Marseille Université (2011)
7. Bailly, R., Denis, F., Ralaivola, L.: Grammatical inference as a principal component analysis problem. In: Proceedings of ICML (2009)
8. Balle, B., Mohri, M.: On the Rademacher complexity of weighted automata. In: Proceedings of ALT (2015)
9. Balle, B.: Learning Finite-State Machines: Statistical and Algorithmic Aspects. Ph.D. thesis, Universitat Politecnica de Catalunya (2013)

10. Balle, B., Carreras, X., Luque, F.M., Quattoni, A.: Spectral learning of weighted automata. Machine learning 96(1-2) (2014)
11. Balle, B., Mohri, M.: Spectral learning of general weighted automata via constrained matrix completion. In: Proceedings of NIPS (2012)
12. Beimel, A., Bergadano, F., Bshouty, N.H., Kushilevitz, E., Varricchio, S.: On the applications of multiplicity automata in learning. In: Proceeding FOCS (1996)
13. Beimel, A., Bergadano, F., Bshouty, N.H., Kushilevitz, E., Varricchio, S.: Learning functions represented as multiplicity automata. Journal of the ACM 47(3) (2000)
14. Bergadano, F., Varricchio, S.: Learning behaviors of automata from multiplicity and equivalence queries. In: Proceedings of CIAC, vol. 778. Springer (1994)
15. Bergadano, F., Varricchio, S.: Learning behaviors of automata from multiplicity and equivalence queries. SIAM Journal on Computing 25(6) (1996)
16. Berstel, J., Reutenauer, C.: Rational Series and Their Languages. Springer (1988)
17. Bisht, L., Bshouty, N.H., Mazzawi, H.: On optimal learning algorithms for multiplicity automata. In: Proceedings of COLT (2006)
18. Breuel, T.M.: The OCRopus open source OCR system. In: Proceedings of IS&T/SPIE (2008)
19. Cardon, A., Crochemore, M.: Détermination de la représentation standard d'une série reconnaissable. ITA 14(4), 371–379 (1980)
20. Carlyle, J.W., Paz, A.: Realizations by stochastic finite automata. J. Comput. Syst. Sci. 5(1) (1971)
21. Chalermsook, P., Laekhanukit, B., Nanongkai, D.: Pre-reduction graph products: Hardnesses of properly learning dfas and approximating edp on dags. In: Proceedings of FOCS (2014)
22. Cortes, C., Haffner, P., Mohri, M.: Rational kernels: Theory and algorithms. Journal of Machine Learning Research 5 (2004)
23. Droste, M., Kuich, W. (eds.): Handbook of weighted automata. EATCS Monographs on Theoretical Computer Science, Springer (2009)
24. Dupont, P., Denis, F., Esposito, Y.: Links between probabilistic automata and hidden markov models: probability distributions, learning models and induction algorithms. Pattern Recognition (2005)
25. Durbin, R., Eddy, S.R., Krogh, A., Mitchison, G.J.: Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids. Cambridge University Press (1998)
26. Eilenberg, S.: Automata, Languages and Machines. Academic Press (1974)
27. Fazel, M.: Matrix rank minimization with applications. Ph.D. thesis, Stanford University (2002)
28. Fliess, M.: Matrices de Hankel. Journal de Mathématiques Pures et Appliquées 53 (1974)
29. Gold, E.M.: Complexity of automaton identification from given data. Information and Control 3(37) (1978)
30. Golub, G., Loan, C.V.: Matrix Computations. Johns Hopkins University Press (1983)
31. Haussler, D., Littlestone, N., Warmuth, M.K.: Predicting $\{0, 1\}$-Functions on Randomly Drawn Points. In: Proceedings of COLT (1988)
32. Hsu, D., Kakade, S.M., Zhang, T.: A spectral algorithm for learning hidden markov models. In: Proceedings of COLT (2009)
33. Hsu, D., Kakade, S.M., Zhang, T.: A spectral algorithm for learning hidden markov models. Journal of Computer and System Sciences 78(5) (2012)
34. II, K.C., Kari, J.: Image compression using weighted finite automata. Computers & Graphics 17(3) (1993)
35. Kaplan, R.M., Kay, M.: Regular models of phonological rule systems. Computational Linguistics 20(3) (1994)
36. Karttunen, L.: The replace operator. In: Proceedings of ACL (1995)
37. Kearns, M.J., Valiant, L.G.: Cryptographic limitations on learning boolean formulae and finite automata. Journal of ACM 41(1) (1994)

38. Kearns, M.J., Vazirani, U.V.: An Introduction to Computational Learning Theory. MIT Press (1994)
39. Kiefer, S., Marusic, I., Worrell, J.: Minimisation of multiplicity tree automata. In: Proceedings of FOSSACS (2015)
40. Kuich, W., Salomaa, A.: Semirings, Automata, Languages. Springer (1986)
41. Mohri, M.: Finite-state transducers in language and speech processing. Computational Linguistics 23(2) (1997)
42. Mohri, M.: Weighted automata algorithms. In: Handbook of Weighted Automata. Springer (2009)
43. Mohri, M., Pereira, F., Riley, M.: Weighted automata in text and speech processing. In: Proceedings of ECAI-96 Workshop on Extended finite state models of language (1996)
44. Mohri, M., Pereira, F., Riley, M.: Speech recognition with weighted finite-state transducers. In: Handbook on Speech Processing and Speech Comm. Springer (2008)
45. Mohri, M., Pereira, F.C.N.: Dynamic compilation of weighted context-free grammars. In: Proceedings of COLING-ACL (1998)
46. Mohri, M., Rostamizadeh, A., Talwalkar, A.: Foundations of Machine Learning. MIT Press (2012)
47. Mohri, M., Sproat, R.: An efficient compiler for weighted rewrite rules. In: Proceedings of ACL (1996)
48. Mornhinweg, D., Shapiro, D.B., Valente, K.: The principal axis theorem over arbitrary fields. American Mathematical Monthly (1993)
49. Pereira, F., Riley, M.: Speech recognition by composition of weighted finite automata. In: Finite-State Language Processing. MIT Press (1997)
50. Pitt, L., Warmuth, M.K.: The minimum consistent DFA problem cannot be approximated within any polynomial. J. ACM 40(1) (1993)
51. Quattoni, A., Balle, B., Carreras, X., Globerson, A.: Spectral regularization for max-margin sequence tagging. In: Proceedings of ICML (2014)
52. Salomaa, A., Soittola, M.: Automata-Theoretic Aspects of Formal Power Series. Springer (1978)
53. Schützenberger, M.P.: On a special class of recurrent events. The Annals of Mathematical Statistics 32(4) (1961)
54. Schützenberger, M.P.: On the definition of a family of automata. Information and Control 4 (1961)
55. Sproat, R.: A finite-state architecture for tokenization and grapheme-to-phoneme conversion in multilingual text analysis. In: Proceedings of the ACL SIGDAT Workshop. ACL (1995)
56. Valiant, L.G.: A theory of the learnable. Commun. ACM 27(11) (1984)
57. Vidal, E., Thollard, F., de la Higuera, C., Casacuberta, F., Carrasco, R.C.: Probabilistic finite-state machines – part I. PAMI (2005)
58. Vidal, E., Thollard, F., de la Higuera, C., Casacuberta, F., Carrasco, R.C.: Probabilistic finite-state machines – part II. PAMI (2005)