JMLR: Workshop and Conference Proceedings vol 40:1–24, 2015

On-Line Learning Algorithms for Path Experts with Non-Additive Losses

Corinna Cortes CORINNA@GOOGLE.COM
Google Research, New York

Vitaly Kuznetsov VITALY@CIMS.NYU.EDU
Courant Institute, New York

Mehryar Mohri MOHRI@CIMS.NYU.EDU
Courant Institute and Google Research, New York

Manfred K. Warmuth MANFRED@UCSC.EDU
University of California at Santa Cruz

Abstract

We consider two broad families of non-additive loss functions covering a large number of applications: *rational losses* and *tropical losses*. We give new algorithms extending the Follow-the-Perturbed-Leader (FPL) algorithm to both of these families of loss functions and similarly give new algorithms extending the Randomized Weighted Majority (RWM) algorithm to both of these families. We prove that the time complexity of our extensions to rational losses of both FPL and RWM is polynomial and present regret bounds for both. We further show that these algorithms can play a critical role in improving performance in applications such as structured prediction.

Keywords: on-line learning, experts, non-additive losses, structured prediction

1. Introduction

On-line learning algorithms are increasingly adopted as the key solution to modern learning applications with very large data sets of several hundred million or billion points. These algorithms process one sample at a time with an update per iteration that is often computationally cheap and easy to implement. As a result, they are substantially more efficient both in time and space than standard batch learning algorithms with optimization costs often prohibitive for large data sets. Furthermore, on-line learning algorithms benefit from a rich theoretical analysis with regret bounds that are often very tight (Cesa-Bianchi and Lugosi, 2006).

A standard learning paradigm in on-line learning is the expert setting. In this setting, the algorithm maintains a distribution over a set of experts (or picks an expert from an implicitly maintained distribution). At each round, the loss assigned to each expert is revealed. The algorithm incurs the expected loss over the experts and then updates his distribution on the set of experts. The objective of the learner is to minimize his expected regret, which is defined as the cumulative loss of the algorithm minus the cumulative loss of the best expert chosen in hindsight.

There are several natural algorithms for this setting. One straightforward algorithm is the so-called Follow-the-Leader (FL) algorithm which consists of selecting at each round the expert with the smallest cumulative loss. However, this algorithm can be shown not to admit a favorable worst case regret. An alternative algorithm with good regret guarantees is the Randomized

Weighted Majority (RWM) (Littlestone and Warmuth, 1994) (and the related Hedge algorithm (Freund and Schapire, 1997)). This algorithm maintains one weight per expert i which is proportional to $\exp(-\eta L_i)$, where L_i is the current total loss of expert i and η a positive learning rate. Thus, in the RWM algorithm, the minimum of the FL algorithm is replaced by a "softmin". An alternative algorithm with similar guarantees is the Follow-the-Perturbed-Leader (FPL) algorithm (Kalai and Vempala, 2005) which first perturbs the total losses of the experts with a properly scaled additive noise and then picks the expert with minimum total perturbed loss. Both of these algorithms have a parameter (the learning rate or the scale factor of the additive noise). These parameters must be tuned in hindsight for the algorithms to achieve the optimal regret bound. More recently, an alternative perturbation was proposed which consists of *dropping* to zero the loss of each expert with probability one half. When FL is applied to the total "dropout" losses, the resulting algorithm achieves the optimum regret without tuning any parameter (van Erven et al., 2014).

Most learning problems arising in applications such as machine translation, automatic speech recognition, optical character recognition, or computer vision admit some structure. In these problems, the experts can be viewed as paths in a directed graph with each edge representing a disjoint sub-structure corresponding to a word, phoneme, character, or image patch. This motivates our study of on-line learning with paths experts. Note that the number of path experts can be exponentially larger than the number of edges in the graph. The learning guarantees of the best algorithms just mentioned remain informative in this context since their dependency on the number of experts is only logarithmic. However, the computational complexity of these algorithms also directly depends on the number of experts, which makes them in general impractical in this context.

This problem has been extensively studied in the special case of additive losses where the loss of a path expert is the sum of the losses of its constituent edges. The *Expanded Hedge* (EH) algorithm of Takimoto and Warmuth (2003), an extension of RWM combined with weight pushing (Mohri, 1997, 2009), and an adaptation of FPL (Kalai and Vempala, 2005) to this context both provide efficient solutions to this problem with a polynomial-time complexity with respect to the size of the expert graph. For these algorithms, the range of the loss appears in the regret guarantee. The Component Hedge (CH) algorithm of Koolen, Warmuth, and Kivinen (2010) provides an alternative solution to this problem with somewhat worse polynomial complexity but regret guarantees independent of the range of the loss that are further shown to be tight.

Unfortunately the relevant losses in structured learning applications such as machine translation, speech recognition, other natural language processing applications, or computational biology are often not simply additive in the constituent edges of the path, they are *non-additive*. In machine translation, the loss is based on the BLEU score similarity, which is essentially defined as the inner product of the vectors of n-gram counts of two sequences typically with $n = 4$. The loss therefore depends on overlapping path segments of typically four consecutive edges. In some computational biology tasks, the losses are based on metrics such as gappy n-gram similarity and other measures based on the sum of the product of the counts of common subsequences between two sequences. In speech recognition and many other language processing applications, or in optical character recognition, the loss is based on an edit-distance with non-uniform edit costs. These are the loss functions used to measure the performance of these systems in practice and are therefore crucial to be able to optimize for. None of the algorithms mentioned above can be applied using such non-additive losses with a polynomial-time complexity.

This paper seeks precisely to address this problem. We introduce two broad classes of non-additive losses that cover most applications encountered in practice: *rational losses* and *tropical* *losses* (Section 4). Rational losses are loss functions expressed in terms of sequence kernels, such as the n-gram loss defining the BLEU score and other more complex losses based on rational kernels (Cortes et al., 2004). They can be represented by weighted transducers over the probability semiring. Tropical losses include the generalized edit-distance loss used in speech recognition and other applications, where typically different costs are associated to insertions, deletions, and substitutions. They can be represented by weighted transducers over the tropical semiring.

We describe a new polynomial-time algorithm, *Follow-the-Perturbed-Rational Leader* (FPRL), extending the FPL algorithm to a broad class of rational losses, which includes n-gram losses and most other kernel-based sequential loss functions used in natural language processing and computational biology (Section 5). Our algorithm is based on weighted automata and transducer algorithms such as composition and weighted determinization (Mohri, 2009). Our proof that the time complexity of the algorithm is polynomial is based on a detailed analysis of the weighted determinization algorithm in this context and makes use of new string combinatorics arguments (Section 5.4). We also prove new regret guarantees for our algorithm since existing bounds cannot be readily used (Section 5.2). We further briefly describe a new algorithm, *Follow-the-Perturbed-Tropical Leader* (FPTL), extending FPL to tropical losses (Appendix B). Remarkably, FPTL differs from FPRL only by a change of semiring.

In Appendix C, we give a new efficient algorithm, *Rational Randomized Weighted-Majority* (RRWM), extending RWM to rational loss functions by leveraging ideas from semiring theory and weighted automata algorithms and prove that our algorithm admits a polynomial-time complexity. We further briefly describe a new algorithm, *Tropical Randomized Weighted-Majority* (TRWM), extending RWM to tropical losses (Appendix D). Remarkably, as for our extensions of FPL, TRWM differs from RRWM only by a change of semiring.

We start with a description of the learning scenario (Section 2) and the introduction of some background material on semirings and transducers (Section 3).

2. Learning scenario

Let $\mathcal{X}$ denote the input space and $\mathcal{Y} = \Sigma^*$ the output space defined as the set of sequences over a finite and non-empty alphabet Σ. We will denote by ϵ the empty string. We consider a standard on-line learning scenario of prediction with expert advice. At each round $t \in [1, T]$, the learner receives an input point $x_t \in \mathcal{X}$ as well as the predictions made by N experts, which he uses to define a probability distribution p_t over $\mathcal{Y}$. He then makes a prediction by drawing $\widehat{y}_t \sim \mathsf{p}_t$, thereby incurring a loss $L(y_t, \widehat{y}_t)$, where $y_t \in \mathcal{Y}$ is the correct label of x_t.

The experts we consider are accepting paths of a finite automaton and their predictions the labels of those paths. For any $t \in [1, T]$, we denote by $\mathcal{A}_t = (\Sigma, Q, I, F, E_t)$ a finite automaton over the alphabet Σ, where Q is a finite set of states, $I \subseteq Q$ the set of initial, $F \subseteq Q$ the set of final states and $E_t \subseteq Q \times (\Sigma \cup \{\epsilon\}) \times Q$ a finite set of (labeled) edges or transitions from state to state. For two distinct rounds t and t', $\mathcal{A}_t$ and $\mathcal{A}_{t'}$ differ only (possibly) by the labels of the transitions. Thus, we can define a path expert ξ as any sequence of edges from I to F and the prediction made by ξ at round t as the label of path ξ in $\mathcal{A}_t$, which we denote by $\xi(t)$. The loss incurred by path expert ξ at round t is $L(y_t, \xi(t))$. We will consider the standard case in applications where the automata $\mathcal{A}_t$ are acyclic, though many of our results apply in the general case of non-acyclic automata. Figure 1(a) shows an example of path expert automaton where the transition labels help us identify path experts.

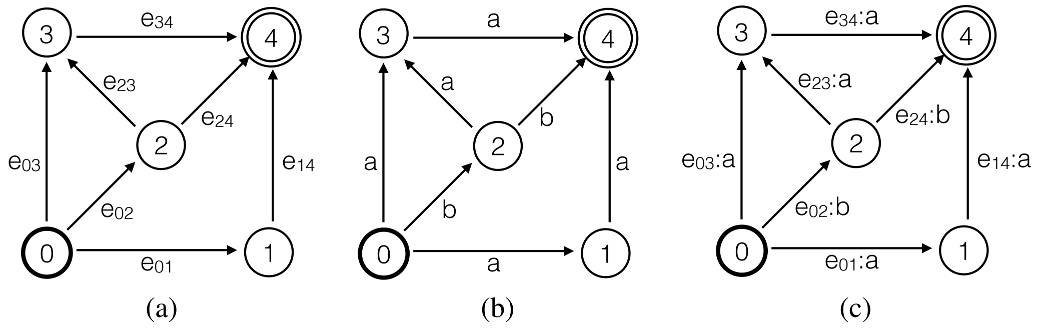


Figure 1: (a) Example of an expert automaton. Each transition is labeled with a distinct symbol of the form e_{pq} encoding the source p and destination state q. The automaton is thus deterministic by construction and each path expert is identified by its sequence of transition labels. For example, $\xi = e_{02}e_{23}e_{34}$ denotes the path expert from 0 to 4 going through states 2 and 3. (b) Example of an automaton $\mathcal{A}_t$. As an example, the prediction of path expert $\xi = e_{02}e_{23}e_{34}$ is $\xi(t) = baa$. (c) Transducer $\mathcal{T}_t$ mapping path experts to their $\mathcal{A}_t$ predictions. For each transition, a colon separates the input symbol from the output predicted label.

Figure 1(b) shows an example of an automaton $\mathcal{A}_t$ defined by the predictions made by path experts. Figure 1(c) shows the transducer $\mathcal{T}_t$ which maps path experts to their predictions in $\mathcal{A}_t$.

In some familiar problems such as the online shortest-path problem, the loss function is simply a sum of the losses over the constituent edges of ξ. But, as already mentioned, in several important applications such as machine translation, speech recognition, pronunciation modeling, and many other natural language processing applications, the loss function is not additive in the edges of the path. We will study two broad families of non-additive losses that cover the loss functions used in all of these applications: *rational losses* and *tropical losses*.

3. Semirings and weighted transducers

The families of non-additive loss functions we consider can be naturally described in terms of *weighted transducers* where the weights are elements of a *semiring*. Thus, in this section, we briefly introduce some basic notions and notation related to semirings and weighted transducers needed for the discussion in the following sections. For a detailed presentation of weighted transducer algorithms and their properties, we refer the reader to (Mohri, 2009).

3.1. Definitions and properties

A weighted finite automaton (WFA) $\mathcal{A}$ is a finite automaton whose transitions and final 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 $a \in \mathbb{S}$.

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. 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. That is $+$ serves as the operation along a path and gives the length of a path, and $\min$ operates on path lengths and determines the smallest one.

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, by $I_{\mathcal{A}} \in Q_{\mathcal{A}}$ its initial state, by $F_{\mathcal{A}} \subseteq Q_{\mathcal{A}}$ its set of final states, and by $E_{\mathcal{A}}$ its finite set of transitions, which are elements of $Q_{\mathcal{T}} \times (\Sigma \cup \{\epsilon\}) \times \mathbb{S} \times Q_{\mathcal{T}}$.[1] We will also denote by $\rho_{\mathcal{A}}(q) \in \mathbb{S}$ the weight of a final state $q \in F_{\mathcal{A}}$ and by $w_{\mathcal{A}}[e] \in \mathbb{S}$ the weight of a transition e. 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.

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

$$\forall x \in \Sigma^*, \quad \mathcal{A}(x) = \bigoplus_{\pi \in P(I_{\mathcal{A}}, x, F_{\mathcal{A}})} \Big(w_{\mathcal{A}}[\pi] \otimes \rho_{\mathcal{A}}(\mathrm{dest}[\pi]) \Big), \tag{1}$$

where $P(I_{\mathcal{A}}, x, F_{\mathcal{A}})$ is the set of paths labeled with x from the initial state to a final state; by convention, $\mathcal{A}(x) = \overline{0}$ when $P(I_{\mathcal{A}}, x, F_{\mathcal{A}}) = \emptyset$. Note that the sum runs over a finite set in the absence of ϵ-cycles. More generally, for all the semirings we consider, the order of the terms in this $\oplus$-sum does not matter, thus the quantity is well defined. 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 $\mathcal{A}$: $|\mathcal{A}| = |Q_{\mathcal{A}}| + |E_{\mathcal{A}}|$.

A weighted finite-state transducer (WFST) $\mathcal{T}$ is a weighted automaton whose transitions are augmented with some output labels which are elements of an output finite alphabet Δ. We will use the same notation and definitions for WFSTs as for WFAs. As for WFAs, a WFST $\mathcal{T}$ over an input alphabet Σ and output alphabet Δ defines a function mapping the pairs of strings in $\Sigma^* \times \delta^*$ to $\mathbb{S}$ that is abusively denoted by $\mathcal{T}$ and defined as follows:

$$\forall x \in \Sigma^*, \forall y \in \Delta^*, \quad \mathcal{T}(x, y) = \bigoplus_{\pi \in P(I_{\mathcal{T}}, x, y, F_{\mathcal{T}})} \Big(w_{\mathcal{T}}[\pi] \otimes \rho_{\mathcal{T}}(\mathrm{dest}[\pi]) \Big), \tag{2}$$

where $P(I_{\mathcal{T}}, x, y, F_{\mathcal{T}})$ is the set of paths from an initial state to a final state with input label x and output label y; $\mathcal{T}(x, y) = \overline{0}$ if $P(I_{\mathcal{T}}, x, y, F_{\mathcal{T}}) = \emptyset$. The *size of a WFST* is defined in the same way as for WFAs. Figure 2 shows some examples.

3.2. Weighted transducer operations

The following are some standard operations defined over WFSTs.

The *inverse* of a WFST $\mathcal{T}$ is denoted by $\mathcal{T}^{-1}$ and defined as the transducer obtained by swapping the input and output labels of every transition of $\mathcal{T}$, thus $\mathcal{T}^{-1}(x, y) = \mathcal{T}(y, x)$ for all (x, y).

1. To simplify the presentation, we are assuming a unique initial state with no weight. This, however, does not represent a restriction since any weighted automaton admits an equivalent weighted automaton of this form. All of our results can be straightforwardly extended to the case of multiple initial states with initial weights and also to the case where $E_{\mathcal{A}}$ is a multiset, thereby allowing multiple transitions between the same two states with the same labels.

The *projection* of a WFST $\mathcal{T}$ is the weighted automaton denoted by $\Pi(\mathcal{T})$ obtained from $\mathcal{T}$ by omitting the input label of each transition and keeping only the output label.

The $\oplus$-*sum* of two WFSTs $\mathcal{T}_1$ and $\mathcal{T}_2$ with the same input and output alphabets is a weighted transducer denoted by $\mathcal{T}_1 \oplus \mathcal{T}_2$ and defined by $(\mathcal{T}_1 \oplus \mathcal{T}_2)(x, y) = \mathcal{T}_1(x, y) \oplus \mathcal{T}_2(x, y)$ for all (x, y). $\mathcal{T}_1 \oplus \mathcal{T}_2$ can be computed from $\mathcal{T}_1$ and $\mathcal{T}_2$ in linear time, that is $O(|\mathcal{T}_1| + |\mathcal{T}_2|)$ where $|\mathcal{T}_1|$ is the sum of the number of states and transitions in the size of $\mathcal{T}_1$ and $\mathcal{T}_2$, simply by merging the initial states of $\mathcal{T}_1$ and $\mathcal{T}_2$. The sum can be similarly defined for WFAs.

The *composition* of two WFSTs $\mathcal{T}_1$ with output alphabet Δ and $\mathcal{T}_2$ with a matching input alphabet Δ is a weighted transducer defined for all x, y by (Pereira and Riley, 1997; Mohri et al., 1996):

$$(\mathcal{T}_1 \circ \mathcal{T}_2)(x, y) = \bigoplus_{z \in \Delta^*} \Big(\mathcal{T}_1(x, z) \otimes \mathcal{T}_2(z, y) \Big). \tag{3}$$

The sum runs over all strings z labeling a path of $\mathcal{T}_1$ on the output side and a path of $\mathcal{T}_2$ on input label z. The worst case complexity of computing $(\mathcal{T}_1 \circ \mathcal{T}_2)$ is quadratic, that is $O(|\mathcal{T}_1||\mathcal{T}_2|)$, assuming that the $\otimes$-operation can be computed in constant time (Mohri, 2009). The composition operation can also be used with WFAs by viewing a WFA as a WFST with equal input and output labels at every transition. Thus, for two WFAs $\mathcal{A}_1$ and $\mathcal{A}_2$, $(\mathcal{A}_1 \circ \mathcal{A}_2)$ is a WFA defined for all x by

$$(\mathcal{A}_1 \circ \mathcal{A}_2)(x) = \mathcal{A}_1(x) \otimes \mathcal{A}_2(x). \tag{4}$$

As for (unweighted) finite automata, there exists a *determinization* algorithm for WFAs. The algorithm returns a deterministic WFA equivalent to its input WFA (Mohri, 1997). Unlike the unweighted case, weighted determinization is not defined for all input WFAs but it can be applied in particular to any acyclic WFA, which is the case of interest for us. When it can be applied to $\mathcal{A}$, we will denote by $\mathrm{Det}(\mathcal{A})$ the deterministic WFA returned by determinization. In Section E.1, we give a brief description of a version of the weighted determinization algorithm.

4. Non-additive losses

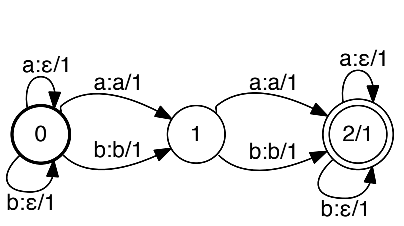
4.1. Rational losses

A *rational kernel* over $\Sigma^* \times \Sigma^*$ is a kernel defined by a weighted transducer $\mathcal{U}$ whose input and output alphabets are both Σ. Thus, $\mathcal{U}(x, x')$ is the value of that kernel for two strings $x, x' \in \Sigma^*$. It was shown by Cortes, Haffner, and Mohri (2004) that any $\mathcal{U}$ of the form $\mathcal{U} = \mathcal{T} \circ \mathcal{T}^{-1}$, where $\mathcal{T}$ is an arbitrary weighted transducer over the probability semiring with matching input and output alphabets, is positive definite symmetric (PDS). Furthermore, the sequence kernels commonly found in computational biology and natural language processing have precisely this form. As an example, Figure 2(a) shows the weighted transducer $\mathcal{T}_{\text{bigram}}$ used to define the bigram kernel $\mathcal{U}_{\text{bigram}}$, which associates to two strings x and x' over the alphabet $\Sigma = \{a, b\}$ the sum over all bigrams of the product of the count of each bigram in x and x'.

Let $\mathcal{U}$ be a weighted transducer over the probability semiring admitting Σ as both the input and output alphabet. Then, we define the *rational loss* associated to $\mathcal{U}$ as the function $L_{\mathcal{U}} \colon \Sigma^* \times \Sigma^* \to \mathbb{R} \cup \{-\infty, +\infty\}$ defined for all $x, y \in \Sigma^*$ by[2]

$$L_{\mathcal{U}}(x, y) = -\log\big(\mathcal{U}(x, y)\big). \tag{5}$$

2. The definition of a rational loss in an earlier version of this paper did not include a logarithm. This new definition is more natural, in particular for kernels based on counts, and, further, leads to a simpler presentation.



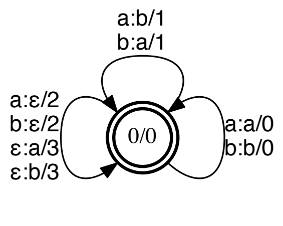


Figure 2: The weight of each transition (or that of a final state) is indicated after the slash separator. (a) Bigram transducer $\mathcal{T}_{\text{bigram}}$ over the probability semiring. A bigram is a sequence of two consecutive symbols, thus aa, ab, ba, or bb for the alphabet $\Sigma = \{a, b\}$. For example, for any string x, $\mathcal{T}_{\text{bigram}}(x, ab)$ is equal to the number of occurrences of ab in x. (a) Edit-distance transducer $\mathcal{U}_{\text{edit}}$ over the tropical semiring, in the case where the substitution cost is 1, the deletion cost 2, the insertion cost 3, and the alphabet $\Sigma = \{a, b\}$.

4.2. Tropical losses

The general edit-distance of two sequences is the minimal cost of a series of edit operations transforming one sequence into the other, where the edit operations are insertion, deletion, and substitution, with non-negative costs not necessarily equal. It is known that the general edit-distance of two sequences x and y can be computed using a weighted transducer $\mathcal{U}_{\text{edit}}$ over the tropical semiring in time $O(|x||y|)$ (Mohri, 2003). Figure 2(b) shows that transducer for an alphabet of size two and some particular edit costs. The edit-distance of two sequences $x, y \in \Sigma^*$ is $\mathcal{U}_{\text{edit}}(x, y)$.

Let $\mathcal{U}$ be a weighted transducer over the tropical semiring admitting Σ as both the input and output alphabet and with $\mathcal{U}(x, y) \geq 0$ for all $x, y \in \Sigma^*$. Then, we define the *tropical loss* associated to $\mathcal{U}$ as the function $L_{\mathcal{U}} \colon \Sigma^* \times \Sigma^* \to \mathbb{R}$ coinciding with $\mathcal{U}$; thus, for all $x, y \in \Sigma^*$,

$$L_{\mathcal{U}}(x, y) = \mathcal{U}(x, y). \tag{6}$$

5. Follow-the-Perturbed-Rational-Leader (FPRL) algorithm

This section describes an on-line learning algorithm for path experts with a rational loss that can be viewed as an extension of FPL (Kalai and Vempala, 2005). We first introduce the definition of several WFSTs and WFA needed for the presentation of the algorithm.

5.1. Description

To any WFST $\mathcal{U}$ over the probability semiring, we associate a corresponding WFST $\widetilde{\mathcal{U}}$ over the log semiring by replacing every transition weight or final weight x of $\mathcal{U}$ with $-\log(x)$. Observe that, by definition, for any $x, y \in \Sigma^*$, we can then write

$$\widetilde{\mathcal{U}}(x, y) = \bigoplus_{\pi \in P(I_{\widetilde{\mathcal{U}}}, x, y, F_{\widetilde{\mathcal{U}}})}{}_{\log} \; w_{\widetilde{\mathcal{U}}}[\pi] = \bigoplus_{\pi \in P(I_{\mathcal{U}}, x, y, F_{\mathcal{U}})}{}_{\log} -\log(w_{\mathcal{U}}[\pi]) = -\log \left[\sum_{\pi \in P(I_{\mathcal{U}}, x, y, F_{\mathcal{U}})} w_{\mathcal{U}}[\pi] \right] = -\log\big(\mathcal{U}(x, y)\big).$$

For any $t \in [1, T]$, we augment the finite automaton $\mathcal{A}_t$ with output labels to help us explicitly keep track of edges and paths. That is, each transition $(p, a, q) \in E_{\mathcal{A}_t}$ of $\mathcal{A}_t$ is replaced with (p, a, e_{pq}, q), where e_{pq} represents the transition between p and q, as illustrated by Figure 1(c). We further assign the weight 0 to every transition and final weights. This results in a WFST $\mathcal{T}_t$ which we interpret over the log semiring and which assigns weight 0 to every pair $(\xi(t), \xi)$.

For any $t \in [1, T]$, let $\mathcal{Y}_t$ denote a simple finite automaton accepting only the sequence y_t. We assign weight 0 to every transition of $\mathcal{Y}_t$ and its final weight. In this way, $\mathcal{Y}_t$ is a WFA over the $\log$ semiring which assigns weight 0 to the sequence y_t, the only sequence it accepts (and weight $+\infty$ to all other sequences).

The following proposition helps guide the design of our algorithm. Its proof is given in Appendix A.

Proposition 1 *Let $L_{\mathcal{U}}$ be a rational loss associated to the WFST $\mathcal{U}$ over the probability semiring. For any $t \in [1, T]$, let $\mathcal{V}_t$ denote the WFA over the* $\log$ *semiring defined by $\mathcal{V}_t = \mathrm{Det}(\Pi(\mathcal{Y}_t \circ \widetilde{\mathcal{U}} \circ \mathcal{T}_t))$ and $\mathcal{W}_t$ the WFA over the* $\log$ *semiring defined by $\mathcal{W}_t = \mathcal{V}_1 \circ \cdots \circ \mathcal{V}_t$. Then, for any $t \in [1, T]$ and any path expert ξ, the following equality holds:*

$$\mathcal{W}_t(\xi) = \sum_{s=1}^{t} L_{\mathcal{U}}(y_s, \xi(s)). \tag{7}$$

Figure 3 gives the pseudocode of our FPRL algorithm. At each round $t \in [1, T]$ the learner receives the input point x_t as well as the predictions made by the path experts, which are represented by the transducer $\mathcal{T}_t$ mapping each path expert ξ to its prediction $\xi(t)$. The deterministic WFA $\mathcal{V}_t$ computed at line 5 gives the loss incurred at time t by each path expert ξ: $\mathcal{V}_t(\xi) = \mathcal{U}(y_t, \xi(t))$, as seen in the proof of Proposition 1. By Proposition 1, the WFA $\mathcal{W}_t$ obtained by composition of $\mathcal{W}_{t-1}$ and $\mathcal{V}_t$ (line 6) gives the cumulative loss up to time t for each path expert ξ: $\mathcal{W}_t(\xi) = \sum_{s=1}^{t} L_{\mathcal{U}}(y_s, \xi(s))$. Furthermore, by induction, $\mathcal{W}_t$ is deterministic: $\mathcal{W}_0$ is deterministic; assume that $\mathcal{W}_{t-1}$ is deterministic, then $\mathcal{W}_t$ is deterministic as the result of the composition of two deterministic WFAs (Lemma 9).

Now, as with the FPL algorithm, our algorithm adds some perturbation to the cumulative loss of each expert. To do so, we define a *noise WFA* $\mathcal{N}_t$ by starting with the graph expert automaton $\mathcal{A}_1$ and augmenting each of its transitions with a weight obtained by sampling from a standard Laplacian distribution scaled by $\frac{1}{\epsilon}$, where ϵ is a parameter of the algorithm (lines 7-9), and by setting the final weights to 0. By construction, $\mathcal{N}_t$ is deterministic since $\mathcal{A}_1$ is deterministic. The perturbations are added simply by composing $\mathcal{W}_t$ with $\mathcal{N}_t$ since by definition of composition in the $\log$ semiring, $(\mathcal{W}_t \circ \mathcal{N}_t)(\xi) = \mathcal{W}_t(\xi) + \mathcal{N}_t(\xi) = \sum_{s=1}^{t} L_{\mathcal{U}}(y_s, \xi(s)) + \mathcal{N}_t(\xi)$, where $\mathcal{N}_t(\xi)$ is the sum of the random Laplacian noises along path ξ in $\mathcal{N}_t$. Since both $\mathcal{W}_t$ and $\mathcal{N}_t$ are deterministic, their composition is also deterministic (Lemma 9). Thus, to determine the expert ξ minimizing $(\mathcal{W}_t \circ \mathcal{N}_t)(\xi) = \sum_{s=1}^{t} L_{\mathcal{U}}(y_s, \xi(s)) + \mathcal{N}_t(\xi)$, it suffices to find the shortest path of $\mathcal{W} = \mathcal{W}_t \circ \mathcal{N}_t$ and returns its label $\widehat{y}_{t+1}$.

Note that our use of weighted determinization in the computation of $\mathcal{V}_t$ is key since it ensures that $\mathcal{V}_t$ and thereby $\mathcal{W}_t$ and $\mathcal{W} = \mathcal{W}_t \circ \mathcal{N}_t$ are deterministic. Otherwise, if multiple paths were labeled by the same path expert in $\mathcal{W}$, then, the problem of determining the path expert with the minimal weight in $\mathcal{W}$ would be hard. Of course, instead of determinizing the WFA defining $\mathcal{V}_t$, we could simply determinize $\mathcal{W}$. But, we will later prove that, while the worst-case complexity of weighted determinization is exponential, its complexity is only polynomial in the context where we are using it. It is not clear if the same property would hold for $\mathcal{W}$.

An alternative method improving our algorithm consists of using, instead of determinization, a recent disambiguation algorithm for WFAs (Mohri and Riley, 2015), which returns an equivalent WFA with no two accepting paths sharing the same label.

```
FPRL($T$)
1  $\mathcal{W}_0 \leftarrow 0$ ▷ deterministic one-state WFA over log semiring mapping all strings to 0.
2  for $t \leftarrow 1$ to $T$ do
3      $x_t \leftarrow$ RECEIVE()
4      $\mathcal{T}_t \leftarrow$ PATHEXPERTPREDICTIONTRANSDUCER($x_t$)
5      $\mathcal{V}_t \leftarrow \mathrm{Det}(\Pi(\mathcal{Y}_t \circ \widetilde{\mathcal{U}} \circ \mathcal{T}_t))$
6      $\mathcal{W}_t \leftarrow \mathcal{W}_{t-1} \circ \mathcal{V}_t$
7      $\mathcal{N}_t \leftarrow \mathcal{A}_1$
8      for $e \in E_{\mathcal{N}_t}$ do
9          $w_{\mathcal{N}_t}[e] \leftarrow$ SAMPLE($\frac{1}{\epsilon}\mathcal{L}$) ▷ standard Laplacian distribution scaled by $\frac{1}{\epsilon}$.
10     $\mathcal{W} \leftarrow \mathcal{W}_t \circ \mathcal{N}_t$
11     $\widehat{y}_{t+1} \leftarrow$ SHORTESTPATH($\mathcal{W}, (\min, +)$)
12 return $\mathcal{W}_T$
```

Figure 3: Pseudocode of FPRL.

5.2. Regret guarantees

In this section, we present a regret guarantee for the FPRL algorithm. This will require new proofs since the analysis of Kalai and Vempala (2005) does not apply in our context as the loss function is non-additive. Furthermore, the FPL bounds of Cesa-Bianchi and Lugosi (2006) for the expert setting do not directly apply to our setting either since the perturbations $\mathcal{N}_t(\xi)$ are correlated.

Observe that the topologies of the WFAs $\mathcal{N}_t$s coincide for all t since they are all derived from $\mathcal{A}_1$ by augmenting transitions with random weights. Let n denote the total number of transitions of $\mathcal{A}_1$, $n = |E_{\mathcal{A}_1}|$. We denote by $p_t \in \mathbb{R}^n$ the perturbation random vector whose components are the transition weights $w_{\mathcal{N}_t}[e]$, for some fixed ordering of the transitions $e \in E_{\mathcal{N}_t}$. Each path ξ can be viewed as a binary vector in $\mathbb{R}^n$ with the non-zero components correspond to its constituent transitions. We will denote by $\mathcal{P} \subseteq \mathbb{R}^n$ the set of all such binary vectors.

Let N denote the number of paths in $\mathcal{A}_1$. For any $t \in [1, T]$, let $\ell_t \in \mathbb{R}^N$ denote the vector of path losses at round t and denote by $\mathcal{L} \subset \mathbb{R}^N$ the set of all such vectors:

$$\mathcal{L} = \left\{\ell_t \in \mathbb{R}^N : \ell_t(\xi) = L_{\mathcal{U}}(y_t, \mathcal{A}_t(\xi)), t \in [1, \ldots, T]\right\},$$

where $\ell_t(\xi)$ is the component of ℓ_t indexed by path expert ξ. Let R be a $N \times n$ matrix where each row is a binary vector representing a path ξ in $\mathcal{A}_1$. Our regret guarantees are in terms of $A = \sup_{\ell \in \mathcal{L}} \|R^\dagger \ell\|_1$. In particular, observe that $A \leq \sup_{\ell \in \mathcal{L}} \|\ell\|_1 \|R^\dagger\|_\infty$.

Theorem 2 *For any sequence $y_1, \ldots, y_T$, the following upper bound holds for the expected regret of FPRL run with parameter $\epsilon \leq \min\left(1, \frac{1}{2A}\right)$:*

$$\mathbb{E}\left[R_T(\mathit{FPRL})\right] \leq 4\sqrt{DAL_{\min}(1 + \log n)} + 16DA(1 + \log n),$$

where $L_{\min}$ is the cumulative loss of the best expert in hindsight, $A = \sup_{\ell \in \mathcal{L}} \|R^\dagger \ell\|_1$, $D = \sup_{d, d' \in \mathcal{P}} \|d - d'\|_1$ and where n is the number of transitions of the path expert automaton $\mathcal{A}_1$.

5.3. Running-time complexity

Since the worst-case complexity of each composition is quadratic, at each round t, $(\mathcal{Y}_t \circ \widetilde{\mathcal{U}} \circ \mathcal{T}_t)$ can be computed in time $O(|\mathcal{U}||\mathcal{A}_1||y_t|)$ using the equalities $|\widetilde{\mathcal{U}}| = |\mathcal{U}|$ and $|\mathcal{T}_t| = |\mathcal{A}_1|$. Similarly, $\mathcal{W}_t$ can be computed in $O(|\mathcal{W}_{t-1}||\mathcal{V}_t|)$ and $\mathcal{W}$ in time $O(|\mathcal{W}_t||\mathcal{N}_t|)$. Since $\mathcal{W}$ is acyclic, its single-source shortest-path (line 11) can be found in linear time, that is $O(|\mathcal{W}|) = O(|\mathcal{W}_t||\mathcal{N}_t|)$. Thus, to show that our algorithm admits a polynomial-time complexity, it suffices to show that the cost of the determinization of $\Pi(\mathcal{Y}_t \circ \widetilde{\mathcal{U}} \circ \mathcal{T}_t)$ (line 5) is also polynomial.[3]

5.4. Determinization of $\Pi(\mathcal{Y}_t \circ \widetilde{\mathcal{U}} \circ \mathcal{T}_t)$

Let $\mathcal{M}$ denote $\mathcal{Y}_t \circ \widetilde{\mathcal{U}} \circ \mathcal{T}_t$. This section analyzes the determinization of $\Pi(\mathcal{M})$ for any t. We will assume here that the number of possible distinct expert predictions $\mathcal{N}$ at any round is polynomial in the size of the automaton $\mathcal{A}_1$, which is a mild assumption clearly holding in all applications that we are aware of – in fact, we will only require that the number of distinct multiplicities of the sequences reaching any state q of $\Pi(\mathcal{M})$ is polynomial in $|\Pi(\mathcal{M})|$.

5.4.1. TRANSDUCERS FOR COUNTING AND COUNTING-BASED RATIONAL LOSSES

Suppose we wish to compute $-\log$ of the number of occurrences of the sequences accepted by a an arbitrary deterministic finite automaton (DFA) Z. In the simple case of bigrams, Z would be a DFA accepting the strings aa, ab, ba, and bb. We can augment Z with weights all equal to 0, including final weights and interpret the resulting automaton Z as one defined over the log semiring. Then, it has been shown by Cortes, Haffner, and Mohri (2004) that the simple WFST $\mathcal{T}_Z$ of Figure 4(a) over the log semiring can be used to compute $(-\log)$ of the number of occurrences of every sequence accepted by Z. That is, for any sequence $x \in \Sigma^*$ and any sequence z accepted by Z, we have $\mathcal{T}_Z(x, z) = -\log(|x|_z)$, where $|x|_z$ denotes the number of occurrences of z in x. This is because the number of distincts decompositions of x into $x_1 z x_2$ with x_1 a prefix of x and x_2 a suffix of x is precisely the number of occurrences of z, and because $\mathcal{T}_Z$ admits exactly one path for each of these decompositions. These paths are obtained by consuming a prefix x_1 of x while remaining at state 0 and outputting ϵ, next reading z to reach state 2, and finally consuming a suffix x_2 at state 2 and outputting ϵ. Since each path has weight zero, the $\oplus_{\log}$ of the weights of these paths is $-\log(|x|_z)$.

Most sequence kernels used in applications, including n-gram kernels, other natural language processing kernels, and kernels used in computational biology and many others are instances of rational kernels of the type $\mathcal{U} = \mathcal{T}_Z \circ \mathcal{T}_Z^{-1}$ (Cortes et al., 2004). In view of that, in what follows, we will assume that $\mathcal{U}$ is of this form, which covers most applications in practice.

5.4.2. CHARACTERIZATION OF THE STATES OF $\mathcal{M} = (\mathcal{Y}_t \circ \widetilde{\mathcal{U}} \circ \mathcal{T}_t)$

Since $\mathcal{U} = \mathcal{T}_Z \circ \mathcal{T}_Z^{-1}$, we can write $\mathcal{M} = \mathcal{Y}_t \circ \widetilde{\mathcal{U}} \circ \mathcal{T}_t = \mathcal{Y}_t \circ \widetilde{\mathcal{T}}_Z \circ \widetilde{\mathcal{T}}_Z^{-1} \circ \mathcal{T}_t$, where $\widetilde{\mathcal{T}}_Z$ is the WFST over the log semiring obtained from $\mathcal{T}$ by replacing each weight x by $(-\log(x))$. Thus, to compute $\mathcal{M}$, we can separately compute $(\mathcal{Y}_t \circ \widetilde{\mathcal{T}}_Z)$ and $(\widetilde{\mathcal{T}}_Z^{-1} \circ \mathcal{T}_t)$ and then compose them.

Recall that a *factor* f of a string y is a sequence of contiguous symbols, possibly empty, appearing in y: thus there exists $y_1, y_2 \in \Sigma^*$ such that $y = y_1 f y_2$. In view of our discussion of a counting transducer $\widetilde{\mathcal{T}}_Z$, by definition of composition, the states of $\mathcal{Y}_t \circ \widetilde{\mathcal{T}}_Z$ are pairs (p, u), where p is the

3. After each determinization step, we can apply weighted minimization (Mohri, 1997) to reduce the size of the resulting WFA. Since the WFAs are acyclic here, the time and space complexity of minimization is only linear in that case.

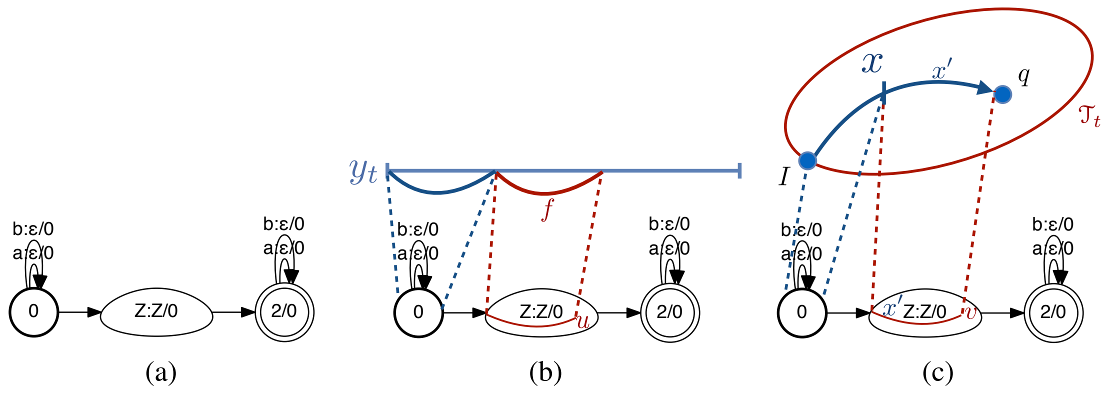


Figure 4: (a) WFST $\widetilde{\mathcal{T}}_Z$ over the log semiring and the alphabet $\Sigma = \{a, b\}$ computing $(-\log)$ of the number of occurrences of sequences accepted by an acyclic DFA Z. $Z{:}Z/0$ represents a WFST over the log semiring obtained from a DFA Z, by augmenting each transition with an output label matching the input label and by setting all weights to be equal to 0. (b) Illustration of the states (p, u) of $\mathcal{Y}_t \circ \widetilde{\mathcal{T}}_Z$. (c) Illustration of the states (v, q) of $\widetilde{\mathcal{T}}_Z^{-1} \circ \widetilde{\mathcal{T}}_t$.

position in y_t at which a factor f of y_t is ending and where u is the unique state of $\widetilde{\mathcal{T}}_Z$ reached by reading f: since Z is deterministic, f cannot reach two distincts states of Z; also, if reading a prefix of f leads to the final state of $\widetilde{\mathcal{T}}_Z$, then f leads to that final state too. Figure 4(b) illustrates this.

Similarly, the states of $\widetilde{\mathcal{T}}_Z^{-1} \circ \mathcal{T}_t$ are precisely the pairs (v, q), where q is a state of $\mathcal{T}_t$ reached by some sequence x, and where v is the (unique) state of $\mathcal{T}_Z$ reached by a suffix x' of x. This is illustrated by Figure 4(c). Now, the transducer $(\mathcal{Y}_t \circ \widetilde{\mathcal{T}}_Z)$ may have ϵ-labels on the output and similarly $(\widetilde{\mathcal{T}}_Z^{-1} \circ \mathcal{T}_t)$ may have ϵ-labels on the input. To avoid the construction of redundant ϵ-paths in the composition $(\mathcal{Y}_t \circ \widetilde{\mathcal{T}}_Z) \circ (\mathcal{T}_Z^{-1} \circ \mathcal{T}_t)$, a *composition filter* is used in the composition algorithm, which itself consists of a simple 3-state transducer (or even 2-state) transducer (Mohri et al., 1996; Mohri, 2009). We will denote by $s \in \{0, 1, 2\}$ the state of that filter transducer.

Thus, the states of $(\mathcal{Y}_t \circ \widetilde{\mathcal{T}}_Z) \circ (\widetilde{\mathcal{T}}_Z^{-1} \circ \mathcal{T}_t)$ are 5-tuples (p, u, s, v, q) where (p, u) and (v, q) are as just described and where, additionally, the factor f of y_t coincides with the suffix x' of x. Thus, they are precisely the 5-tuples where p is the ending position of a factor f of y_t, q the state reached by some string x in the input of $\mathcal{T}_t$ that admits f as a suffix, u the state reached by reading f in $\widetilde{\mathcal{T}}_Z$ and v the one reached by reading $x' = f$ on the output of $\widetilde{\mathcal{T}}_Z^{-1}$, and s one of three filter states.

5.4.3. CHARACTERIZATION OF THE STATES AND SUBSETS CONSTRUCTED BY $\mathrm{Det}(\Pi(\mathcal{M}))$

Theorem 3 *The complexity of computing* $\mathrm{Det}(\Pi(\mathcal{M}))$*, the determinization of* $\Pi(\mathcal{M}) = \Pi(\mathcal{Y}_t \circ \mathcal{U} \circ \mathcal{T}_t)$*, is in* $O(\mathcal{N}|\mathcal{U}||y_t||\mathcal{A}_1|)$.

Proof The application of determinization to $\Pi(\mathcal{M})$ creates states defined by weighted subsets. Let S be such a weighted subset reached by a path ξ. S contains states of the form $(p_i, u_i, s_i, v_i, q_i)$, $i = 1, \ldots, n$, which are all reached by the same path ξ. But, since the path expert automaton $\mathcal{A}_t$ is deterministic, the states q_i must be all equal to the unique state q reached in $\mathcal{A}_t$ by path ξ. Thus, the subsets of S are all of the form (p_i, u_i, s_i, v_i, q) and we will refer to them as q-*subsets*.

We now further analyze the different possible q-subsets constructed by weighted determinization. In view of the characterizations previously given for 5-tuples of the form (p, u, s, v, q) ending with state q, all 5-tuples of a q-subset S are defined by a string x reaching q on the input side of $\mathcal{T}_t$, and the different factors of y_t matching a suffix of x, each creating a new element of S (with

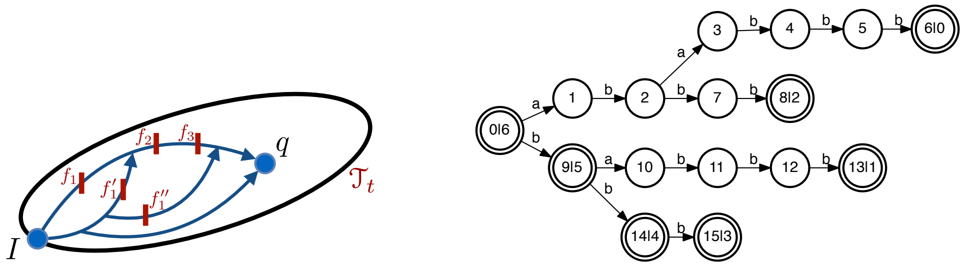


Figure 5: (a) Illustration of sequences of factors of y_t, $f_1 \geq_s f_2 \geq_s \cdots \geq_s f_n$, along the set of paths with destination state q in $\mathcal{T}_t$. Each factor f_i is indicated in red by its starting position along a path, its ending positing being always state q. (b) Suffix trie of $y_t = ababbb$. The number of sequences $f_1 \geq_s f_2 \geq_s \cdots \geq_s f_n$ is the number of paths to a final state, that is the number of positions in y_t (marked at each final state after the separator $|$) .

possibly multiple filter states s). Thus, two q-subsets differ precisely by the set of factors f in y_t that are suffixes of a sequence ending in q. How many different sequences of factors can be suffixes of the set of sequences x ending in q? Figure 5(a) illustrates the situation.

For any two strings z_1 and z_2, we write $z_1 \geq_s z_2$ when z_2 is a suffix of z_1. Observe that if $f_1, \ldots, f_n$ is the sequence of factors in y_t, suffixes of a sequence x ending at q, then we must have $f_1 \geq_s f_2 \geq_s \cdots \geq_s f_n$. Thus, the number of distinct q-subsets is the number of different sequences of factors of y_t ordered for the suffix order. The number of such sequences is at most $|y_t| + 1$ as it can be seen straightforwardly from a suffix trie representing y_t. Figure 5(b) illustrates that.

This proves that for any given state q of $\mathcal{A}_t$, the number of distinct unweighted q-subsets constructed by the determinization of $\mathrm{Det}(\Pi(\mathcal{Y}_t \circ \mathcal{U} \circ \mathcal{T}_t))$ is at most $O(|y_t||\mathcal{U}|)$. Thus, the total number of distinct subsets is at most $O(|y_t||\mathcal{U}||\mathcal{A}_t|)$.

We now examine the number of possible remainder weights for each unweighted subset. Consider any q-subset that is reached by a path labeled with x. If there is another q-subset containing exactly the same tuples with different weights, then there must be another path labeled with x_0 that goes to the same q-subset. Note, however, that, by definition of determinization, if the number of paths labeled with x_0 with destination q in $\Pi(\mathcal{M})$ is the same as the number of paths labeled with x and with destination q in $\Pi(\mathcal{M})$ then this will result in the same remainders and will not create new states. It follows that the number of possible remainders is precisely the number of counts of different strings labeling paths ending in state q of $\mathcal{W}_t$, which is bounded by $\mathcal{N}$. ■

6. Conclusion

We presented algorithms extending FPL and RWM to the setting of path experts when using two important families of non-additive loss functions. These constitute significant advances in on-line learning broadening the applicability of these algorithms to critical applications such as machine translation, automatic speech recognition, and computational biology. In Appendix F, we describe the application of these algorithms and their extensions to several tasks, including machine translation and automatic speech recognition.

Acknowledgments

This work was partly funded by the NSF award IIS-1117591 (Mohri), the NSERC PGS D3 award (Kuznetsov) and the NSF award IIS-1118028 (Warmuth).

References

C. Allauzen and M. Mohri. Efficient algorithms for testing the twins property. *Journal of Automata, Languages and Combinatorics*, 8(2):117–144, 2003.

C. Allauzen, M. Riley, J. Schalkwyk, W. Skut, and M. Mohri. OpenFst: a general and efficient weighted finite-state transducer library. In *Proceedings of CIAA*, pages 11–23. Springer, 2007.

N. Cesa-Bianchi and G. Lugosi. *Prediction, learning, and games*. Cambridge University Press, 2006.

C. Cortes, P. Haffner, and M. Mohri. Rational kernels: Theory and algorithms. *JMLR*, 5, 2004.

C. Cortes, V. Kuznetsov, and M. Mohri. Ensemble methods for structured prediction. In *Proceedings of ICML*, 2014.

Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

A. Kalai and S. Vempala. Efficient algorithms for online decision problems. *Journal of Computer and System Sciences*, 71(3):291–307, 2005.

W. M. Koolen, M. K. Warmuth, and J. Kivinen. Hedging structured concepts. In *Proceedings of COLT*, pages 93–105, 2010.

N. Littlestone and M. K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2):212–261, 1994.

M. Mohri. Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2):269–311, 1997.

M. Mohri. Edit-distance of weighted automata: General definitions and algorithms. *Int. J. Found. Comput. Sci.*, 14(6):957–982, 2003.

M. Mohri. Weighted automata algorithms. In *Handbook of Weighted Automata*, pages 213–254. Springer, 2009.

M. Mohri and M. Riley. On the disambiguation of weighted automata. In *Proceedings of CIAA*. Springer, 2015.

M. Mohri, F. Pereira, and M. Riley. Weighted automata in text and speech processing. In *Proceedings of ECAI-96 Workshop on Extended finite state models of language*, 1996.

M. Mohri, F. Pereira, and M. Riley. A rational design for a weighted finite-state transducer library. In *Proceedings of WIA*, volume 1436 of *LNCS*, pages 144–158. Springer, 1998.

F. Pereira and M. Riley. Speech recognition by composition of weighted finite automata. In *Finite-State Language Processing*, pages 431–453. MIT Press, 1997.

E. Takimoto and M. K. Warmuth. Path kernels and multiplicative updates. *JMLR*, 4:773–818, 2003.

T. van Erven, W. Kotlowski, and M. K. Warmuth. Follow the leader with dropout perturbations. In *COLT*, 2014.

Appendix A. FPRL proofs

Proposition 1 *Let $L_{\mathcal{U}}$ be a rational loss associated to the WFST $\mathcal{U}$ over the probability semiring. For any $t \in [1, T]$, let $\mathcal{V}_t$ denote the WFA over the* log *semiring defined by $\mathcal{V}_t = \mathrm{Det}(\Pi(\mathcal{Y}_t \circ \widetilde{\mathcal{U}} \circ \mathcal{T}_t))$ and $\mathcal{W}_t$ the WFA over the* log *semiring defined by $\mathcal{W}_t = \mathcal{V}_1 \circ \cdots \circ \mathcal{V}_t$. Then, for any $t \in [1, T]$ and any path expert ξ, the following equality holds:*

$$\mathcal{W}_t(\xi) = \sum_{s=1}^{t} L_{\mathcal{U}}(y_s, \xi(s)). \tag{8}$$

Proof By definition of composition, for any $s \in [1, T]$ and any path expert ξ, $\Pi(\mathcal{Y}_s \circ \widetilde{\mathcal{U}} \circ \mathcal{T}_s)(\xi)$ can be expressed as follows:

$$\Pi(\mathcal{Y}_s \circ \widetilde{\mathcal{U}} \circ \mathcal{T}_s)(\xi) = \bigoplus_{z_1, z_2}{}_{\log} \Big(\mathcal{Y}_s(z_1) + \widetilde{\mathcal{U}}(z_1, z_2) + \mathcal{T}_s(z_2, \xi)\Big).$$

Since $\mathcal{Y}_s$ only accepts the sequence y_s, the sum can be restricted to $z_1 = y_s$. Similarly, since $\mathcal{T}_s$ admits only one path with output ξ, the one with input label $\xi(s)$, the sum can be restricted to $z_2 = \xi(s)$. Thus, the expression simplifies into the following:

$$\Pi(\mathcal{Y}_s \circ \widetilde{\mathcal{U}} \circ \mathcal{T}_s)(\xi) = \mathcal{Y}_s(y_s) + \widetilde{\mathcal{U}}(y_s, \xi(s)) + \mathcal{T}_s(\xi(s), \xi) = \widetilde{\mathcal{U}}(y_s, \xi(s)) = L_{\mathcal{U}}(y_s, \xi(s)),$$

using the fact that, by construction, we have $\mathcal{Y}_s(y_s) = 0$ and $\mathcal{T}_s(\xi(s), \xi) = 0$. The WFA $\mathcal{V}_s$ is equivalent to $\Pi(\mathcal{Y}_s \circ \mathcal{U} \circ \mathcal{T}_s)$ since it is obtained from it via determinization. Thus, we also have $\mathcal{V}_s(\xi) = \mathcal{U}(y_s, \xi(s))$ for any path expert ξ. Now, by definition of composition, we can write

$$\mathcal{W}_t(\xi) = \mathcal{V}_1(\xi) + \cdots + \mathcal{V}_t(\xi) = \sum_{s=1}^{t} L_{\mathcal{U}}(y_s, \xi(s)),$$

which concludes the proof. ■

We will adopt the following notation, which is similar to the one used in the proofs for FPL: for any WFA $\mathcal{A}$, $M(\mathcal{A}) = \operatorname{argmin}_\xi \mathcal{A}(\xi)$. Using this notation, the following analogue of a standard result in FPL proofs holds (Kalai and Vempala, 2005; Cesa-Bianchi and Lugosi, 2006).

Lemma 4 *Let L be a loss function such that for all $t \in [1, T]$, $\sum_{s=1}^{t} L(y_s, \xi(s)) = \mathcal{X}_t(\xi)$ for some WFAs $\mathcal{X}_1, \ldots, \mathcal{X}_T$. Then, the following holds:*

$$\sum_{t=1}^{T} L(y_t, M(\mathcal{X}_t)(t)) \leq \sum_{t=1}^{T} L(y_t, M(\mathcal{X}_T)(t)).$$

Proof The proof is by induction on T. The inequality clearly holds for $T = 1$. Now, assume that it holds for $t \leq T - 1$, then

$$\sum_{t=1}^{T} L(y_t, M(\mathcal{X}_t)(t)) \leq \sum_{t=1}^{T-1} L(y_t, M(\mathcal{X}_{T-1})(t)) + L(y_T, M(\mathcal{X}_T)(T)).$$

By the definition of $M(\mathcal{X}_{T-1})$ as a minimizer, we have

$$\sum_{t=1}^{T-1} L(y_t, M(\mathcal{X}_{T-1})(t)) \leq \sum_{t=1}^{T-1} L(y_t, M(\mathcal{X}_T)(t)).$$

Thus, we can write

$$\sum_{t=1}^{T} L(y_t, M(\mathcal{X}_t)(t)) \leq \sum_{t=1}^{T-1} L(y_t, M(\mathcal{X}_T)(t)) + L(y_T, M(\mathcal{X}_T)(T)) = \sum_{t=1}^{T} L(y_t, M(\mathcal{X}_T)(t)),$$

which completes the proof. ∎

Lemma 5 *For any any sequence of vectors $y_1, \ldots, y_T$, the following holds:*

$$\sum_{t=1}^{T} L_{\mathcal{U}}(y_t, M(\mathcal{W}_t \circ \mathcal{N}_t)(t)) \leq \sum_{t=1}^{T} L_{\mathcal{U}}(y_t, M(\mathcal{W}_T)(t)) + D \sum_{t=1}^{T} \|p_t - p_{t-1}\|_\infty,$$

where $D = \sup_{d,d' \in \mathcal{P}} \|d - d'\|_1$.

Proof For any path ξ, let $\overline{\xi}$ denote the binary vector in $\mathbb{R}^n$ representing ξ. Let $p_0 = 0$ and consider the loss function L such that for $t \in [1, T]$, $L(y_t, \xi(t)) = L_{\mathcal{U}}(y_t, \xi(t)) + \overline{\xi} \cdot (p_t - p_{t-1})$. Observe that

$$\sum_{s=1}^{t} L(y_t, \xi(t)) = \sum_{s=1}^{t} L_{\mathcal{U}}(y_t, \xi(s)) + \sum_{s=1}^{t} \overline{\xi} \cdot (p_s - p_{s-1}) = \sum_{s=1}^{t} L_{\mathcal{U}}(y_t, \xi(t)) + \overline{\xi} \cdot p_t = (\mathcal{W}_t \circ \mathcal{N}_t)(\xi).$$

Thus, we can apply Lemma 4 with the loss function L, which gives

$$\sum_{t=1}^{T} L(y_t, M(\mathcal{W}_t \circ \mathcal{N}_t)(t)) \leq \sum_{t=1}^{T} L(y_t, M(\mathcal{W}_T \circ \mathcal{N}_T)(t)) \leq \sum_{t=1}^{T} L(y_t, M(\mathcal{W}_T)(t)),$$

where we used the minimizing property of $M(\mathcal{W}_T \circ \mathcal{N}_T)$. The inequality relating the first and third term can be written explicitly as follows:

$$\sum_{t=1}^{T} \Big(L_{\mathcal{U}}(y_t, M(\mathcal{W}_t \circ \mathcal{N}_t)) + \overline{M(\mathcal{W}_t \circ \mathcal{N}_t)} \cdot (p_t - p_{t-1}) \Big) \leq$$

$$\sum_{t=1}^{T} L_{\mathcal{U}}(y_t, M(\mathcal{W}_T)) + \sum_{t=1}^{T} \overline{M(\mathcal{W}_T)} \cdot (p_t - p_{t-1}).$$

Rearranging, we obtain

$$\sum_{t=1}^{T} L_{\mathcal{U}}(M(\mathcal{W}_t \circ \mathcal{N}_t), y_t) \leq \sum_{t=1}^{T} L_{\mathcal{U}}(M(\mathcal{W}_T), y_t) + \sum_{t=1}^{T} (\overline{M(\mathcal{W}_T)} - \overline{M(\mathcal{W}_t \circ \mathcal{N}_t)}) \cdot (p_t - p_{t-1}).$$

The result then follows by application of Hölder's inequality. ∎

Theorem 2 *For any sequence $y_1, \ldots, y_T$, the following upper bound holds for the expected regret of FPRL run with parameter $\epsilon \leq \min\left(1, \frac{1}{2A}\right)$:*

$$\mathbb{E}\left[R_T(\textit{FPRL})\right] \leq 4\sqrt{DAL_{\min}(1+\log n)} + 16DA(1+\log n),$$

where $L_{\min}$ is the cumulative loss of the best expert in hindsight, $A = \sup_{\ell \in \mathcal{L}} \|R^\dagger \ell\|_1$, $D = \sup_{d,d' \in \mathcal{P}} \|d - d'\|_1$ and where n is the number of transitions of the path expert automaton $\mathcal{A}_1$.

Proof Our proof is based on the bound of Lemma 5. We may assume that $p_1 = p_2 = \ldots = p_T$ since only marginal distributions of these random vectors affect the expected regret. Then the bound of Lemma 5 reduces to

$$\sum_{t=1}^{T} L_{\mathcal{U}}(y_t, M(\mathcal{W}_t \circ \mathcal{N}_t)) \leq \sum_{t=1}^{T} L_{\mathcal{U}}(y_t, M(\mathcal{W}_T)) + D\|p_1\|_\infty. \tag{9}$$

Since p_1 is a vector of n independent standard Laplacian random variables scaled by $1/\epsilon$, the following upper bound holds:

$$\mathbb{E}[\|p_1\|_\infty] \leq 2\frac{1+\log n}{\epsilon}. \tag{10}$$

To conclude the proof we need to relate $\mathbb{E}[L_{\mathcal{U}}(M(\mathcal{W}_t \circ \mathcal{N}_1), y_t)]$ and $\mathbb{E}[L_{\mathcal{U}}(M(\mathcal{W}_{t-1} \circ \mathcal{N}_1), y_t)]$. We will show that the following inequality holds for all t:

$$\mathbb{E}[L_{\mathcal{U}}(y_t, M(\mathcal{W}_{t-1} \circ \mathcal{N}_1))] \leq \exp(\epsilon A)\, \mathbb{E}[L_{\mathcal{U}}(y_t, M(\mathcal{W}_t \circ \mathcal{N}_1))]. \tag{11}$$

Indeed, let $f_{\mathbf{Y}}$ be the joint density of Y_ξs. Observe that $\mathbf{Y} = R\mathbf{X}$, where $\mathbf{Y}$ is a vector of Y_ξ, $\mathbf{X}$ is a vector of n independent Laplacian random variables corresponding to edge noise and R is a matrix where each row is a binary vector representing a path ξ in expert automaton. By change of variables, $f_{\mathbf{Y}}(\mathbf{x}) = f_{\mathbf{X}}(R^\dagger \mathbf{x})|\det^*(R^\dagger)|$, where $R^\dagger$ is a pseudo-inverse of R, $\det^*$ is the pseudo-determinant and $f_{\mathbf{X}}$ is joint density of $\mathbf{X}$. Then it follows that

$$\begin{aligned}
\mathbb{E}[L_{\mathcal{U}}(y_t, M(\mathcal{W}_{t-1} \circ \mathcal{N}_1))] &= \int L_{\mathcal{U}}(y_t, M(\mathcal{W}_{t-1}(\xi) + L_{\mathcal{U}}(\xi, y_t) + x_\xi)) f_{\mathbf{Y}}(\mathbf{x}) d\mathbf{x} \\
&\leq \left(\sup \frac{f_{\mathbf{Y}}(\mathbf{x})}{f_{\mathbf{Y}}(\mathbf{x} - \ell)}\right) \int L_{\mathcal{U}}(M(\mathcal{W}_t(\xi) + x_\xi), y_t) f_{\mathbf{Y}}(\mathbf{x}) d\mathbf{x} \\
&= \left(\sup \frac{f_{\mathbf{Y}}(\mathbf{x})}{f_{\mathbf{Y}}(\mathbf{x} - \ell)}\right) \mathbb{E}[L_{\mathcal{U}}(M(\mathcal{W}_t \circ \mathcal{N}_1), y_t)].
\end{aligned}$$

By definition of $f_{\mathbf{Y}}$ and $f_{\mathbf{X}}$ and the triangle inequality, the following inequality holds:

$$\left(\sup \frac{f_{\mathbf{Y}}(\mathbf{x})}{f_{\mathbf{Y}}(\mathbf{x} - \ell)}\right) \leq \exp\left(\epsilon\|R^\dagger x - R^\dagger \ell\|_1 - \epsilon\|R^\dagger x\|_1\right) \leq \exp(\epsilon A),$$

which shows (11) for all t. Now, summing up the inequalities (11) over t and using (9) and (10) yields

$$\sum_{t=1}^{T} \mathbb{E}[L_{\mathcal{U}}(M(\mathcal{W}_{t-1} \circ \mathcal{N}_t), y_t)] \leq \exp(\epsilon A)\left[L_{\min} + 2D\frac{1+\log n}{\epsilon}\right].$$

For $\epsilon \leq 1$ we have that

$$\sum_{t=1}^{T} \mathbb{E}[L_{\mathcal{U}}(y_t, M(\mathcal{W}_{t-1} \circ \mathcal{N}_t))] \leq (1 + 2\epsilon A)\left[L_{\min} + 2D\frac{1 + \log n}{\epsilon}\right],$$

where $L_{\min}$ is the cumulative loss of the best expert in hindsight. Using $\epsilon = \epsilon_1/2A$ with $\epsilon_1 \leq 1$ leads to the following bound on the expected regret:

$$\sum_{t=1}^{T} \mathbb{E}[L_{\mathcal{U}}(y_t, M(\mathcal{W}_{t-1} \circ \mathcal{N}_t))] - L_{\min} \leq \epsilon_1 L_{\min} + 8D\frac{A(1 + \log n)}{\epsilon_1}.$$

If $8DA(1 + \log n) \leq L_{\min}$, then $\epsilon_1 = \sqrt{\frac{8DA(1+\log n)}{L_{\min}}}$ minimizes the right-hand side and gives the minimizing value $4\sqrt{DAL_{\min}(1 + \log n)}$. Otherwise, $L_{\min} < 8DA(1 + \log n)$ and for $\epsilon_1 = 1$, the right-hand side is bounded by $16DA(1 + \log n)$. ■

Appendix B. Follow-the-Perturbed-Tropical-Leader (FPTL) algorithm

Here, we briefly discuss our on-line learning algorithm for path experts with a tropical loss, the *Follow-the-Perturbed-Tropical-Leader* (FPTL) algorithm, which, as in the case of FPTL, can be viewed as an extension of FPL (Kalai and Vempala, 2005). FPTL syntactically coincides with FPTL (same pseudocode) but, everywhere in the algorithm, instead of the log semiring, the tropical semiring is used. The correctness of the algorithm, including the proof of an analogue of Proposition 1 is straightforward to show using similar techniques. In particular, the algorithm can be used for on-line learning with the edit-distance loss. We are leaving to a future study, however, the key question of the complexity of determinization in FPTL.

Appendix C. Rational Randomized Weighted-Majority (RRWM) algorithm

In this section, we present an efficient algorithm for on-line learning for path experts with a rational loss, Rational Randomized Weighted-Majority (RRWM), which can be viewed as an extension of RWM Littlestone and Warmuth (1994).

C.1. Description

In the context of path experts, the randomized weighted majority (RWM) algorithm (Littlestone and Warmuth, 1994) can be described as follows: (1) first, compute $\exp(-\eta \sum_{t=1}^{T} L(\xi(t), y_t))$ for each path expert ξ; (2) next, sample according to the distribution defined by these weights. This raises several issues: (1) the number of path experts ξ is exponential in the number of states of the automaton, therefore, a naive computation of the cumulative loss per path expert is not practical; (2) even if we compute a WFA whose weight for each path expert ξ is $\exp(-\eta \sum_{t=1}^{T} L(\xi(t), y_t))$, it is not clear how to use that WFA for sampling path experts.

We will show that an elegant algorithmic solution can be derived for these problems by introducing a new semiring and by using some standard WFST algorithms.

Lemma 6 *For any $\eta > 0$, the system $\mathbb{S}_\eta = (\mathbb{R}_+ \cup \{+\infty\}, \oplus_\eta, \times, 0, 1)$, where $\oplus_\eta$ is defined for all $x, y \in \mathbb{R}_+ \cup \{+\infty\}$ by $x \oplus_\eta y = \big(x^{\frac{1}{\eta}} + y^{\frac{1}{\eta}}\big)^\eta$, is a semiring that we will call the* η-power semiring. *Furthermore, the mapping $\Psi_\eta \colon (\mathbb{R}_+ \cup \{+\infty\}, +, \times, 0, 1) \to \mathbb{S}_\eta$ defined by $\Psi_\eta(x) = x^\eta$ is a semiring isomorphism.*

Proof By definition of $\mathbb{S}_\eta$, for all $x, y \in \mathbb{R}_+ \cup \{+\infty\}$, $\Psi_\eta(0) = 0$, $\Psi_\eta(1) = 1$, and

$$\Psi_\eta(x + y) = \Psi_\eta(x) \oplus_\eta \Psi_\eta(y) \quad \text{and} \quad \Psi_\eta(xy) = \Psi_\eta(x)\Psi_\eta(y).$$

Furthermore, $\Psi_\eta(\mathbb{R}_+ \cup \{+\infty\}) = \mathbb{R}_+ \cup \{+\infty\}$. This shows that $\mathbb{S}_\eta$ is a semiring and that Ψ_η is a semiring morphism. It is straightforward to verify that its inverse Ψ_η^{-1} defined by $\Psi_\eta(x) = x^{\frac{1}{\eta}}$ for all $x \in \mathbb{R}_+ \cup \{+\infty\}$ is also a morphism. ■

To any WFST $\mathcal{U}$ over the probability semiring, we associate a corresponding WFST $\widetilde{\mathcal{U}}$ over the η-power semiring by replacing every transition weight or final weight x of $\mathcal{U}$ with x^η. Observe that, by definition, for any $x, y \in \Sigma^*$, we can then write

$$\begin{aligned}
\widetilde{\mathcal{U}}(x, y) &= \bigoplus\nolimits_{\eta}\limits_{\pi \in P(I_{\widetilde{\mathcal{U}}}, x, y, F_{\widetilde{\mathcal{U}}})} w_{\widetilde{\mathcal{U}}}[\pi] \\
&= \Bigg[\sum_{\pi \in P(I_{\widetilde{\mathcal{U}}}, x, y, F_{\widetilde{\mathcal{U}}})} w_{\widetilde{\mathcal{U}}}^{\frac{1}{\eta}}[\pi]\Bigg]^\eta = \Bigg[\sum_{\pi \in P(I_{\mathcal{U}}, x, y, F_{\mathcal{U}})} w_{\mathcal{U}}[\pi]\Bigg]^\eta \\
&= \mathcal{U}(x, y)^\eta = e^{-\eta[-\log \mathcal{U}(x,y)]} = e^{-\eta L_{\mathcal{U}}(x,y)}.
\end{aligned}$$

For any $t \in [1, T]$, we augment the finite automaton $\mathcal{A}_t$ with output labels to help us explicitly keep track of edges and paths. That is, each transition $(p, a, q) \in E_{\mathcal{A}_t}$ of $\mathcal{A}_t$ is replaced with (p, a, e_{pq}, q), where e_{pq} represents the transition between p and q, as illustrated by Figure 1(c). We further assign the weight 1 to every transition and final weights. This results in a WFST $\mathcal{T}_t$ which we interpret over the η-power semiring and which assigns weight 1 to every pair $(\xi(t), \xi)$.

For any $t \in [1, T]$, let $\mathcal{Y}_t$ denote a simple finite automaton accepting only the sequence y_t. We assign weight 1 to every transition of $\mathcal{Y}_t$ and its final weight. In this way, $\mathcal{Y}_t$ is a WFA over the semiring $\mathbb{S}_\eta$ which assigns weight 1 to the sequence y_t, the only sequence it accepts (and weight 0 to all other sequences).

Proposition 7 *Let $L_{\mathcal{U}}$ be a rational loss associated to the WFST $\mathcal{U}$ over the probability semiring. For any $t \in [1, T]$, let $\mathcal{V}_t$ denote the WFA over the semiring $\mathbb{S}_\eta$ defined by $\mathcal{V}_t = \mathrm{Det}(\Pi(\mathcal{Y}_t \circ \widetilde{\mathcal{U}} \circ \mathcal{T}_t))$ and $\mathcal{W}_t$ the WFA over the semiring $\mathbb{S}_\eta$ defined by $\mathcal{W}_t = \mathcal{V}_1 \circ \cdots \circ \mathcal{V}_t$. Then, for any $t \in [1, T]$ and any path expert ξ, the following equality holds:*

$$\mathcal{W}_t(\xi) = e^{-\eta \sum_{s=1}^t L_{\mathcal{U}}(y_s, \xi(s))}. \tag{12}$$

Proof By definition of composition, for any $t \in [1, T]$, $\Pi(\mathcal{Y}_t \circ \widetilde{\mathcal{U}} \circ \mathcal{T}_t)(\xi)$ can be expressed as follows:

$$\Pi(\mathcal{Y}_t \circ \widetilde{\mathcal{U}} \circ \mathcal{T}_t)(\xi) = \bigoplus\nolimits_{\eta}\limits_{z_1, z_2} \Big(\mathcal{Y}_t(z_1)\widetilde{\mathcal{U}}(z_1, z_2)\mathcal{T}_t(z_2, \xi)\Big).$$

Since $\mathcal{Y}_t$ only accepts the sequence y_t, the $\oplus_\eta$-sum can be restricted to $z_1 = y_t$. Similarly, since $\mathcal{T}_t$ admits only one path with output ξ, the one with input label $\xi(t)$, the $\oplus_\eta$-sum can be restricted

```
RRWM($T$)
1  $\mathcal{W}_0 \leftarrow 1$  ▷ deterministic one-state WFA over the semiring $\mathbb{S}_\eta$ mapping all strings to $1$.
2  for $t \leftarrow 1$ to $T$ do
3      $x_t \leftarrow$ RECEIVE()
4      $\mathcal{T}_t \leftarrow$ PATHEXPERTPREDICTIONTRANSDUCER($x_t$)
5      $\mathcal{V}_t \leftarrow \mathrm{Det}(\Pi(\mathcal{Y}_t \circ \widetilde{\mathcal{U}} \circ \mathcal{T}_t))$
6      $\mathcal{W}_t \leftarrow \mathcal{W}_{t-1} \circ \mathcal{V}_t$
7      $\mathcal{W}_t \leftarrow$ WEIGHTPUSH($\mathcal{W}_t, (+, \times)$)
8      $\widehat{y}_{t+1} \leftarrow$ SAMPLE($\mathcal{W}_t$)
9  return $\mathcal{W}_T$
```

Figure 6: Pseudocode of RRWM.

to $z_2 = \xi(t)$. Further, using the fact that, by construction, $\mathcal{Y}_t(y_t) = 1$ and $\mathcal{T}_t(\xi(t), \xi) = 1$, the expression simplifies into the following:

$$\Pi(\mathcal{Y}_t \circ \widetilde{\mathcal{U}} \circ \mathcal{T}_t)(\xi) = \widetilde{\mathcal{U}}(y_t, \xi(t)) = e^{-\eta L_{\mathcal{U}}(y_t, \xi(t))}.$$

The WFA $\mathcal{V}_s$ is equivalent to $\Pi(\mathcal{Y}_s \circ \mathcal{U} \circ \mathcal{T}_s)$ since it is obtained from it via determinization. Thus, we also have $\mathcal{V}_s(\xi) = \mathcal{U}(y_s, \xi(s))$ for any path expert ξ. Now, by definition of composition, we can write

$$\mathcal{W}_t(\xi) = \mathcal{V}_1(\xi) \cdots \mathcal{V}_t(\xi) = e^{-\eta \sum_{s=1}^{t} L_{\mathcal{U}}(y_s, \xi(s))},$$

which concludes the proof. ■

Figure 6 gives the pseudocode of our Rational Randomized Weighted-Majority (RRWM) algorithm. At each round $t \in [1, T]$ the learner receives the input point x_t as well as the predictions made by the path experts, which are represented by the transducer $\mathcal{T}_t$ mapping each path expert ξ to its prediction $\xi(t)$. The deterministic WFA $\mathcal{V}_t$ computed at line 5 gives the loss incurred at time t by each path expert ξ: $\mathcal{V}_t(\xi) = \mathcal{U}(y_t, \xi(t))$, as seen in the proof of Proposition 7. By Proposition 7, the WFA $\mathcal{W}_t$ obtained by composition of $\mathcal{W}_{t-1}$ and $\mathcal{V}_t$ (line 6) gives the cumulative loss up to time t for each path expert ξ: $\mathcal{W}_t(\xi) = \sum_{s=1}^{t} L_{\mathcal{U}}(y_s, \xi(s))$. Furthermore, by induction, $\mathcal{W}_t$ is deterministic: $\mathcal{W}_0$ is deterministic; assume that $\mathcal{W}_{t-1}$, then $\mathcal{W}_t$ is deterministic as the result of the composition of two deterministic WFAs (Lemma 9).

$\mathcal{W}_t(\xi)$ gives the loss of path expert ξ precisely as in the RWM algorithm. To use $\mathcal{W}_t$ for sampling, we need to compute a WFA equivalent to $\mathcal{W}_t$ that is *stochastic* in the probability semiring, that is one that assigns precisely the same weight to each path expert but such that outgoing transition weights at each state sum to one.

The weight pushing algorithm (Mohri, 1997, 2009) can precisely help us compute such a WFA. But, to do so, we would need to apply it to a WFA over the probability semiring while $\mathcal{W}_t$ is defined over $\mathbb{S}_\eta$. Observe however that $\mathbb{S}_\eta$ and the probability semiring share the same times operation (standard multiplication). Furthermore, since $\mathcal{W}_t$ is deterministic, the plus operation of $\mathbb{S}_\eta$ is not needed in the computation of $\mathcal{W}_t(\xi)$ for any path expert ξ. Thus, we can equivalently interpret $\mathcal{W}_t$

as a WFA over the probability semiring. Note that our use of weighted determinization is key to ensure this property.[4]

We now briefly describe the weight pushing algorithm. For any state $q \in Q_{\mathcal{W}_t}$, let $d[q]$ denote the sum of the weights of all paths from q to final states:

$$d[q] = \sum_{\pi \in P(q, F_{\mathcal{W}_t})} w_{\mathcal{W}_t}[\pi] \, \rho_{\mathcal{W}_t}(\text{dest}(\pi)),$$

where $P(q, F_{\mathcal{W}_t})$ denotes the set of paths from q to a state in $F_{\mathcal{W}_t}$. The weight pushing algorithm then consists of the following steps. For any transition $e \in E_{\mathcal{W}_t}$ such that $d[\text{orig}(e)] \neq 0$, update its weight as follows:

$$w_{\mathcal{W}_t}[e] \leftarrow d[\text{orig}(e)]^{-1} \, w_{\mathcal{W}_t}[e] \, d[\text{dest}(e)].$$

For any final state $q \in F_{\mathcal{W}_t}$, update its weight as follows:

$$\rho_{\mathcal{W}_t}[e] \leftarrow \rho_{\mathcal{W}_t}[q] \, d[I_{\mathcal{W}_t}].$$

The resulting WFA is guaranteed to preserve the path expert weights and to be stochastic (Mohri, 2009).

C.2. Regret guarantees

The following regret guarantee for RRWM follows directly the existing bound for RWM (Littlestone and Warmuth, 1994; Cesa-Bianchi and Lugosi, 2006).

Theorem 8 *Let $\mathcal{N}$ be the total number of path experts and M an upper bound on the loss of any path expert. Then, the following upper bound holds for the regret of RRWM:*

$$\mathbb{E}[R_T(\textit{RRWM})] \leq 2M\sqrt{T \log \mathcal{N}}.$$

C.3. Running-time complexity

Since the worst-case complexity of each composition is quadratic, at each round t, $(\mathcal{Y}_t \circ \widetilde{\mathcal{U}} \circ \mathcal{T}_t)$ can be computed in time $O(|\mathcal{U}||\mathcal{A}_1||y_t|)$ using the equalities $|\widetilde{\mathcal{U}}| = |\mathcal{U}|$ and $|\mathcal{T}_t| = |\mathcal{A}_1|$. Similarly, $\mathcal{W}_t$ can be computed in $O(|\mathcal{W}_{t-1}||\mathcal{V}_t|)$. Since $\mathcal{W}_t$ is acyclic, the sums $d[q]$ can be all computed in linear time $O(|\mathcal{W}_t|)$ (Mohri, 1997, 2009). Thus, to show that our algorithm admits a polynomial-time complexity, it suffices to show that the cost of the determinization of $\Pi(\mathcal{Y}_t \circ \widetilde{\mathcal{U}} \circ \mathcal{T}_t)$ (line 5) is also polynomial.[5] This can be shown in the same way as in Theorem 3.

4. As indicated in the discussion of the FPRL algorithm, an alternative method improving RRWM consists of using, instead of determinization, a recent disambiguation algorithm for WFAs (Mohri and Riley, 2015), which returns an equivalent WFA with no two accepting paths sharing the same label.
5. After each determinization step, we can apply weighted minimization (Mohri, 1997) to reduce the size of the resulting WFA. Since the WFAs are acyclic here, the time and space complexity of minimization is only linear in that case.

Appendix D. Tropical Randomized Weighted-Majority (TRWM) algorithm

In this section, we present an efficient algorithm for on-line learning for path experts with a rational loss, Tropical Randomized Weighted-Majority (TRWM), which can be viewed as an extension of RWM (Littlestone and Warmuth, 1994).

It is straightforward to verify that $\Psi_\eta \colon x \mapsto x^\eta$ defines a semiring isomorphism between $(\mathbb{R}_+ \cup \{+\infty\}, \min, +, 0, 1)$ and $(\mathbb{R}_+ \cup \{+\infty\}, \min, \times, 0, 1)$. TRWM syntactically coincides with RRWM (same pseudocode, same definition of $\mathcal{Y}_t$, $\widetilde{\mathcal{U}}$, and $\mathcal{T}_t$), with the difference that, everywhere, instead of the semiring $\mathbb{S}_\eta$, the semiring $(\mathbb{R}_+ \cup \{+\infty\}, \min, \times, 0, 1)$ is used. The correctness of the algorithm, including the proof of an analogue of Proposition 7 is straightforward to show using similar techniques. In particular, note that, since the second operation of the semiring is the standard multiplication, the weight pushing algorithm can be applied to $\mathcal{W}_t$ to derive a stochastic WFA that can used for sampling. Our TRWM algorithm can be used for on-line learning with the edit-distance loss. We are leaving to a future study, however, the key question of the complexity of determinization used in TRWM.

Appendix E. WFA algorithms and properties

E.1. Weighted determinization

Here we give a brief description of a version of the weighted determinization (Mohri, 1997) in the context of the probability semiring $(\mathbb{R}_+ \cup \{+\infty\}, +, \times, 0, 1)$. The algorithm takes as input a WFA $\mathcal{A}$ and returns an equivalent WFA $\mathcal{B}$ that is deterministic: at each state of B no two outgoing transitions share the same label. We will assume here that the weights in $\mathcal{A}$ are positive, otherwise the corresponding transitions or final states could be first removed in linear time. Unlike the familiar unweighted case, not all weighted automata are determinizable (Mohri, 1997; Allauzen and Mohri, 2003) but all acyclic weighted automata are determinizable, which covers the cases of interest for this paper.

Figure 7 illustrates the algorithm in a simple case. The states of the determinized automaton $\mathcal{B}$ are weighted subsets of the form $\{(q_1, w_1), \ldots, (q_n, w_n)\}$, where q_is, $i = 1 \ldots n$, are states of $\mathcal{A}$ reachable from an initial state of $\mathcal{A}$ via the same string x and w_is *remainder weights* (real numbers) adjusted as follows: let w be the weight of a path in $\mathcal{B}$ labeled with x, starting at the initial state and ending at state $\{(q_1, w_1), \ldots, (q_n, w_n)\}$, then $w \times w_i$ is exactly the sum of the weights of all paths labeled with x and ending at q_i in $\mathcal{A}$.

For any set of states U of $\mathcal{A}$ and any string $x \in \Sigma^*$, let $\delta_{\mathcal{A}}(U, x)$ denote the set of states reached from U by paths labeled with x. Also, for any string $x \in \Sigma^*$, let $W(x, q)$ denote the sum of the weights of all paths from $I_{\mathcal{A}}$ to q in $\mathcal{A}$ labeled with x, and and define $W(x)$ as the minimum $\min_q W(x, q)$. We will also denote by $w(q, a, q')$ the sum of the weights of all transitions from q to q' in $\mathcal{A}$ labeled with $a \in \Sigma$.

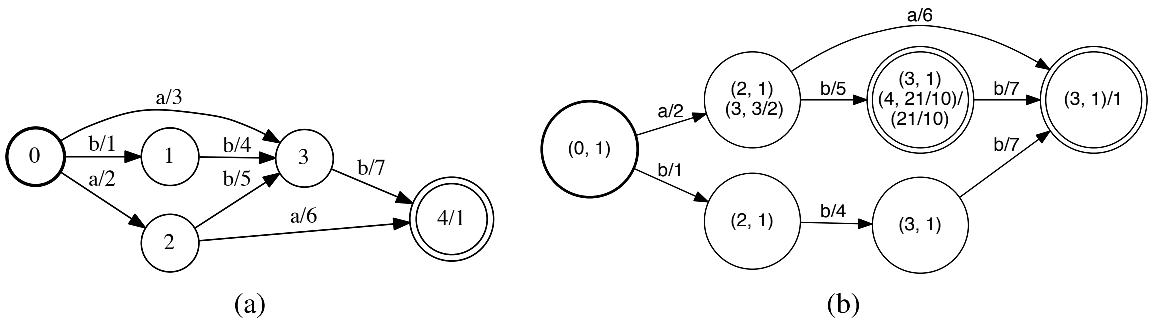


Figure 7: Illustration of weighted determinization. (a) Non-deterministic weighted automaton over the probability semiring $\mathcal{A}$; (b) Equivalent deterministic weighted automaton $\mathcal{B}$. For each state, the pairs elements of the subset are indicated. For final states, the final weight is indicated after the slash separator.

With these definitions, the deterministic weighted automaton $\mathcal{B} = (\Sigma, Q_\mathcal{B}, I_\mathcal{B}, F_\mathcal{B}, E_\mathcal{B}, 1, \rho_\mathcal{B})$, is defined as follows:

$$Q_\mathcal{B} = \left\{ \{(q_1, w_1), \ldots, (q_n, w_n)\} \colon \exists x \in \Sigma^* \mid q_i \in \delta_\mathcal{A}(I_\mathcal{A}, x), w_i = \frac{W(x, q_i)}{W(x)} \right\}$$

$$E_\mathcal{B} = \Big\{ (\mathsf{q}, a, w, \mathsf{q}) \colon \mathsf{q} = \{(q_1, w_1), \ldots, (q_n, w_n)\}, \mathsf{q}' = \{(q'_1, w'_1), \ldots, (q'_r, w'_r)\} \in Q,$$
$$a \in \Sigma, w \in \mathbb{R} \colon \delta_\mathcal{A}\big(\{q_1, \ldots, q_n\}, a\big) = \{q'_1, \ldots, q'_r\}, w = \min(w_i \times w(q_i, a, q'_j))$$
$$w'_j = \frac{w_i \times w(q_i, a, q'_j)}{w} \Big\}$$

$$I_\mathcal{B} = \{(q, 1) : q \in I_\mathcal{A}\}.$$
$$F_\mathcal{B} = \{\mathsf{q} = \{(q_1, w_1), \ldots, (q_n, w_n)\} \in Q \mid \exists i \in [1, n] \colon q_i \in F_\mathcal{A}\},$$

with the final weight function defined by:

$$\forall \mathsf{q} = \{(q_1, w_1), \ldots, (q_n, w_n)\} \in F_\mathcal{B}, \rho_\mathcal{B}(\mathsf{q}) = \sum_{i \colon q_i \in F_\mathcal{A}} w_i.$$

Note that the weighted determinization algorithm defined by the subset construction just described applies also to weighted automata containing ϵ-transitions since, for any set of states U, $\delta_\mathcal{A}(U, x)$ is the set of states reached from U via paths labeled with x that may contain ϵs.

E.2. Composition of deterministic WFAs

Lemma 9 *Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be two deterministic WFSTs over a semiring $(\mathbb{S}, \oplus, \otimes, \overline{0}, \overline{1})$ with the output alphabet of $\mathcal{T}_1$ matching the input alphabet of $\mathcal{T}_2$. Then, the WFST $(\mathcal{T}_1 \circ \mathcal{T}_2)$ returned by the composition algorithm is deterministic.*

Proof The states of $(\mathcal{T}_1 \circ \mathcal{T}_2)$ created by composition can be identified with pairs (p_1, p_2) with $p_1 \in Q_{\mathcal{T}_1}$ and $p_2 \in Q_{\mathcal{T}_2}$. The transitions labeled with a symbol a leaving (p_1, p_2) are obtained by pairing up a transition leaving p_1 and labeled with a with a transition leaving p_2 and labeled with

a. Since $\mathcal{T}_1$ and $\mathcal{T}_2$ are deterministic, there are at most one such transition for p_1 and at most one for p_2. Thus, this leads to the creation of at most one transition leaving (p_1, p_2) and labeled with a, which concludes the proof. ■

E.3. Determinization of a sum of deterministic WFAs

Theorem 10 *Let $\mathcal{A}$ and $\mathcal{B}$ be two acyclic deterministic weighted automata with positive integer transition and final weights defined over the probability semiring. Let $N_\mathcal{A}$ ($N_\mathcal{B}$) denote the largest weight of a path in $\mathcal{A}$ (resp. $\mathcal{B}$), then $\mathrm{Det}(\mathcal{A} + \mathcal{B})$ can be computed in $O(N_\mathcal{A} N_\mathcal{B} |\mathcal{A}||\mathcal{B}|)$.*

Proof Since $\mathcal{A}$ and $\mathcal{B}$ have both a single initial state and are acyclic, a weighted automaton $\mathcal{A} + \mathcal{B}$ can be constructed from $\mathcal{A}$ and $\mathcal{B}$ simply by merging their initial states. By definition of weighted determinization (see Section E.1), the states of $\mathrm{Det}(\mathcal{A} + \mathcal{B})$ can be identified with subsets of states of $\mathcal{A} + \mathcal{B}$ augmented with some remainder weights. Let p be a state of $\mathrm{Det}(\mathcal{A} + \mathcal{B})$ and let π be a path reaching p that starts at the initial state and is labeled with some string ξ. Since both $\mathcal{A}$ and $\mathcal{B}$ are deterministic, there exists at most one path $\pi_\mathcal{A}$ in $\mathcal{A}$ and at most one path $\pi_\mathcal{B}$ in $\mathcal{B}$ labeled with ξ. Therefore, $\mathcal{A} + \mathcal{B}$ admits at most two paths ($\pi_\mathcal{A}$ and $\pi_\mathcal{B}$) that are labeled with ξ. Let $q_\mathcal{A}$ be the destination state of path $\pi_\mathcal{A}$ in $\mathcal{A}$ (when it exists) and $\pi_\mathcal{B}$ the destination of $\pi_\mathcal{B}$ in $\mathcal{B}$. If only $\pi_\mathcal{A}$ exists, then, by definition of weighted determinization we have $p = \{(q_\mathcal{A}, 1)\}$. Similarly, if only $\pi_\mathcal{B}$ exists, then we have $p = \{(q_\mathcal{B}, 1)\}$. If both exist, then, by definition of weighted determinization, we must have $p = \{(q_\mathcal{A}, w_\mathcal{A}), (q_\mathcal{B}, w_\mathcal{B})\}$, where $w_\mathcal{A}$ and $w_\mathcal{B}$ are remainder weights defined by

$$w_\mathcal{A} = \frac{w[\pi_\mathcal{A}]}{\min\{w[\pi_\mathcal{A}], w[\pi_\mathcal{B}]\}} \qquad w_\mathcal{B} = \frac{w[\pi_\mathcal{B}]}{\min\{w[\pi_\mathcal{A}], w[\pi_\mathcal{B}]\}}.$$

In view of that and the fact that $w[\pi_\mathcal{A}] \in [1, N_\mathcal{A}]$ and $w[\pi_\mathcal{B}] \in [1, N_\mathcal{B}]$, there are at most $N_\mathcal{A} N_\mathcal{B}$ distinct possible pairs of remainder weights $(w_\mathcal{A}, w_\mathcal{B})$. The total number of states in $\mathrm{Det}(\mathcal{A}+\mathcal{B})$ is thus at most $O(N_\mathcal{A} N_\mathcal{B} |Q_\mathcal{A}||Q_\mathcal{B}|)$. The number of outgoing transitions of state $p = \{(q_\mathcal{A}, w_\mathcal{A}), (q_\mathcal{B}, w_\mathcal{B})\}$ is at most $(|E[q_\mathcal{A}]| + |E[q_\mathcal{B}]|)$ where $E[q_\mathcal{A}]$ ($E[q_\mathcal{B}]$) is the set of outgoing transitions of $q_\mathcal{A}$ (resp. $q_\mathcal{B}$). Thus, the total number of transitions of $\mathrm{Det}(\mathcal{A} + \mathcal{B})$ is in $O(N_\mathcal{A} N_\mathcal{B} |E_\mathcal{A}||E_\mathcal{B}|)$ and the size of $\mathrm{Det}(\mathcal{A} + \mathcal{B})$ is $O(N_\mathcal{A} N_\mathcal{B}(|Q_\mathcal{A}||Q_\mathcal{B}| + |E_\mathcal{A}||E_\mathcal{B}|)) = O(N_\mathcal{A} N_\mathcal{B} |\mathcal{A}||\mathcal{B}|)$. ■

Appendix F. Application to ensemble structured prediction

The algorithms we discussed in the previous sections can be used to learn ensembles of structured prediction rules, thereby significantly improving the performance of algorithms in a number of areas including machine translation, speech recognition, other language processing areas, optical character recognition, and computer vision.

Structured prediction problems are common in all of these areas. In structured prediction tasks, the label or output associated to an input $x \in \mathcal{X}$ is a structure $y \in \mathcal{Y}$ that can be decomposed and represented by l substructures $y^1, \ldots, y^l$. For example, for the text-to-phonemes pronunciation task in speech recognition, x is a specific word or word sequence and y is its phonemic transcription with y^k chosen to be individual phonemes or more generally n-grams of consecutive phonemes.

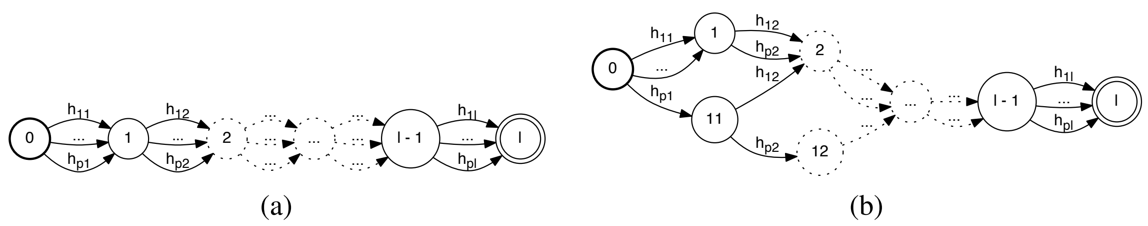


Figure 8: (a) Full automaton of path experts. (b) Arbitrary automaton of path experts.

The problem of learning ensembles of structured prediction rules can be described as follows. As in standard supervised learning problems, the learner receives a training sample m labeled points $S = ((\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_m, \mathbf{y}_m)) \in \mathcal{X} \times \mathcal{Y}$ drawn i.i.d. according to some distribution $\mathcal{D}$ used both for training and testing. The learner is further assumed to have access to a set of p predictors $h_1, \ldots, h_p$ mapping $\mathcal{X}$ to $\mathcal{Y}$ to devise an accurate ensemble prediction. Thus, for any input $\mathbf{x} \in \mathcal{X}$, he can use the prediction of the p experts $h_1(\mathbf{x}), \ldots, h_p(\mathbf{x})$. Each expert predicts l substructures $h_j(x) = (h_j^1(x), \ldots, h_j^l(x))$. Thus, the experts induce multiple path experts, as illustrated by Figures 8. Figure 8(a) shows an automaton of path experts where all combinations are allowed, Figure 8(b) an automaton with a subset of all possible path experts.

The objective of the learner is to use the training sample S to determine the best path expert, that is the combination of substructure predictors with the best expected loss. This is motivated by the fact that one particular expert may be better at predicting the kth substructure while some other expert may be more accurate at predicting another substructure. Therefore, it is desirable to combine the substructure predictions of all experts to derive the more accurate prediction.

It was shown by Cortes, Kuznetsov, and Mohri (2014) that this problem can be efficiently solved using the WMWP or FPL algorithm in the special case of the additive Hamming loss. However, the loss functions used in the applications already listed are non-additive.

We can use the on-line algorithms discussed in the previous sections to derive efficient solutions for the ensemble structured prediction in machine translation with a rational loss is used (BLEU score) or the speech recognition problem where an edit-distance with non-uniform edit costs is used. This requires converting the on-line solutions discussed in the previous section to batch solutions. We present several on-line-to-batch conversion schemes to derive such batch solutions for which we will provide theoretical guarantees.

These algorithms can be implemented using the FSM Library or OpenFst software library (Mohri et al., 1998; Allauzen et al., 2007). We can run experiment with them in a number of machine translation and speech recognition tasks to demonstrate their benefit for the improvement of the accuracy in these tasks. Often, several models are available in these applications, typically as the result of alternative structured prediction learning algorithms, or the result of the training the same algorithm with different features or subsets of the data (for example different gender models in speech recognition). These models can serve as the experts in our formulation and we can seek to derive a solution surpassing their performance by combining the substructures induced by these experts.