Multitask Spectral Learning of Weighted Automata

Guillaume Rabusseau*[1], Borja Balle†‡[2], and Joelle Pineau§[1]

[1] Reasoning and Learning Lab, School of Computer Science, McGill University
[2]Amazon Research Cambridge

June 8, 2017

Abstract

We consider the problem of estimating multiple related functions computed by weighted automata (WFA). We first present a natural notion of relatedness between WFAs by considering to which extent several WFAs can share a common underlying representation. We then introduce the novel model of vector-valued WFA which conveniently helps us formalize this notion of relatedness. Finally, we propose a spectral learning algorithm for vector-valued WFAs to tackle the multitask learning problem. By jointly learning multiple tasks in the form of a vector-valued WFA, our algorithm enforces the discovery of a representation space shared between tasks. The benefits of the proposed multitask approach are theoretically motivated and showcased through experiments on both synthetic and real world datasets.

1 Introduction

One common task in machine learning consists in estimating an unknown function $f : \mathcal{X} \to \mathcal{Y}$ from a training sample of input-output data $\{(x_i, y_i)\}_{i=1}^N$ where each $y_i \simeq f(x_i)$ is a (possibly noisy) estimate of $f(x_i)$. In *multitask learning*, the learner is given several such learning tasks $f_1, \cdots, f_m$. It has been shown, both experimentally and theoretically, that learning related tasks simultaneously can lead to better performances relative to learning each task independently (see e.g. [1, 5], and references therein). Multitask learning has proven particularly useful when few data points are available for each task, or when it is difficult or costly to collect data for a target task while much data is available for related tasks (see e.g. [24] for an example in healthcare). In this paper, we propose a multitask learning algorithm for the case where the input space $\mathcal{X}$ consists of *sequence data*.

Many tasks in natural language processing, computational biology, or reinforcement learning, rely on estimating functions mapping sequences of observations to real numbers: e.g. inferring probability distributions over sentences in language modeling or learning the dynamics of a model of the environment in reinforcement learning. In this case, the function f to infer from training data is defined over the set Σ^* of strings built on a finite alphabet Σ. *Weighted finite automata* (WFA) are finite state machines that allow one to succinctly represent such functions. In particular, WFAs can compute any probability distribution defined by a hidden Markov model (HMM) [9] and can model the transition and observation behavior of partially observable Markov decision processes [22]. A recent line of work has led to the development of *spectral methods* for learning HMMs [14], WFAs [2, 3] and related models, offering an alternative to EM based algorithms with the benefits of being computationally efficient and providing consistent estimators.

*guillaume.rabusseau@mail.mcgill.ca
†pigem@amazon.co.uk
‡This work was done while Borja Balle was lecturer at Lancaster University
§jpineau@cs.mcgill.ca

We consider the problem of multitask learning for WFAs. The notion of relatedness between tasks can be expressed in different ways; one common assumption in multitask learning is that the multiple tasks share a *common underlying representation* [4, 8]. In this paper, we present a natural notion of shared representation between functions defined over strings and we propose a learning algorithm that encourages the discovery of this shared representation. Intuitively, our notion of relatedness captures to which extent several functions can be computed by WFAs sharing a joint forward feature map. In order to formalize this notion of relatedness, we introduce the novel model of *vector-valued WFA* (vv-WFA) which generalizes WFAs to vector-valued functions and offer a natural framework to formalize the multitask learning problem. Given m tasks $f_1, \cdots, f_m : \Sigma^* \to \mathbb{R}$, we consider the function $\vec{f} = [f_1, \cdots, f_m] : \Sigma^* \to \mathbb{R}^m$ whose output for a given input string x is the m-dimensional vector having entries $f_i(x)$ for $i = 1, \cdots, m$. We show that the notion of *minimal vv-WFA* computing $\vec{f}$ exactly captures our notion of relatedness between tasks and we prove that the dimension of such a minimal representation is equal to the rank of a flattening of the *Hankel tensor* of $\vec{f}$ (Theorem 3). Leveraging this result, we design a spectral learning algorithm for vv-WFAs which constitutes a sound multitask learning algorithm for WFAs: by learning $\vec{f}$ in the form of a vv-WFA, rather than independently learning a WFA for each task f_i, we implicitly enforce the discovery of a joint feature space shared among all tasks. After giving a theoretical insight on the benefits of this multitask approach (by leveraging a recent result on asymmetric bounds for singular subspace estimation [6]), we conclude by showcasing these benefits with experiments on both synthetic and real world data.

Related work. Multitask learning for sequence data has previously received limited attention. In [13], mixtures of Markov chains are used to model dynamic user profiles. Tackling the multitask problem with nonparametric Bayesian methods is investigated in [12] to model related time series with Beta processes and in [19] to discover relationships between related datasets using nested Dirichlet process and infinite HMMs. Extending recurrent neural networks to the multitask setting has also recently received some interest (see e.g. [17, 18]). To the best of our knowledge, this paper constitutes the first attempt to tackle the multitask problem for the class of functions computed by general WFAs.

2 Preliminaries

We first present notions on weighted automata, spectral learning of weighted automata and tensors. We start by introducing some notation. We denote by Σ^* the set of strings on a finite alphabet Σ. The empty string is denoted by λ and the length of a string x by $|x|$. For any integer k we let $[k] = \{1, 2, \cdots, k\}$. We use lower case bold letters for vectors (e.g. $\mathbf{v} \in \mathbb{R}^{d_1}$), upper case bold letters for matrices (e.g. $\mathbf{M} \in \mathbb{R}^{d_1 \times d_2}$) and bold calligraphic letters for higher order tensors (e.g. $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{d_1 \times d_2 \times d_3}$). The ith row (resp. column) of a matrix $\mathbf{M}$ will be denoted by $\mathbf{M}_{i,:}$ (resp. $\mathbf{M}_{:,i}$). This notation is extended to slices of a tensor in the straightforward way. Given a matrix $\mathbf{M} \in \mathbb{R}^{d_1 \times d_2}$, we denote by $\mathbf{M}^\dagger$ its Moore-Penrose pseudo-inverse and by $\mathrm{vec}(\mathbf{M}) \in \mathbb{R}^{d_1 d_2}$ its vectorization.

Weighted finite automaton. A *weighted finite automaton* (WFA) with n states is a tuple $A = (\boldsymbol{\alpha}, \{\mathbf{A}^\sigma\}_{\sigma \in \Sigma}, \boldsymbol{\omega})$ where $\boldsymbol{\alpha}, \boldsymbol{\omega} \in \mathbb{R}^n$ are the initial and final weights vectors respectively, and $\mathbf{A}^\sigma \in \mathbb{R}^{n \times n}$ is the transition matrix for each symbol $\sigma \in \Sigma$. A WFA computes a function $f_A : \Sigma^* \to \mathbb{R}$ defined for each word $x = x_1 x_2 \cdots x_k \in \Sigma^*$ by $f_A(x) = \boldsymbol{\alpha}^\top \mathbf{A}^{x_1} \mathbf{A}^{x_2} \cdots \mathbf{A}^{x_k} \boldsymbol{\omega}$.

By letting $\mathbf{A}^x = \mathbf{A}^{x_1} \mathbf{A}^{x_2} \cdots \mathbf{A}^{x_k}$ for any word $x = x_1 x_2 \cdots x_k \in \Sigma^*$ we will often use the shorter notation $f_A(x) = \boldsymbol{\alpha}^\top \mathbf{A}^x \boldsymbol{\omega}$. A WFA A with n states is *minimal* if its number of states is minimal, i.e. any WFA B such that $f_A = f_B$ has at least n states. A function $f : \Sigma^* \to \mathbb{R}$ is *recognizable* if it can be computed by a WFA. In this case the *rank* of f is the number of states of a minimal WFA computing f, if f is not recognizable we let $\mathrm{rank}(f) = \infty$.

Hankel matrix. The *Hankel matrix* $\mathbf{H}_f \in \mathbb{R}^{\Sigma^* \times \Sigma^*}$ associated with a function $f : \Sigma^* \to \mathbb{R}$ is the infinite matrix with entries $(\mathbf{H}_f)_{u,v} = f(uv)$ for $u, v \in \Sigma^*$. The spectral learning algorithm for WFAs relies on the following fundamental relation between the rank of f and the rank of $\mathbf{H}_f$.

Theorem 1. *[7, 11] For any function $f : \Sigma^* \to \mathbb{R}$, $\operatorname{rank}(f) = \operatorname{rank}(\mathbf{H}_f)$.*

Spectral learning. Showing that the rank of the Hankel matrix is upper bounded by the rank of f is easy: given a WFA $A = (\boldsymbol{\alpha}, \{\mathbf{A}^\sigma\}_{\sigma\in\Sigma}, \boldsymbol{\omega})$ with n states, we have the rank n factorization $\mathbf{H}_f = \mathbf{PS}$ where the matrices $\mathbf{P} \in \mathbb{R}^{\Sigma^*\times n}$ and $\mathbf{S} \in \mathbb{R}^{n\times\Sigma^*}$ are defined by $\mathbf{P}_{u,:} = \boldsymbol{\alpha}^\top\mathbf{A}^u$ and $\mathbf{S}_{:,v} = \mathbf{A}^v\boldsymbol{\omega}$ for all $u, v \in \Sigma^*$. The converse is more tedious to show but its proof is constructive, in the sense that it allows one to build a WFA computing f from any rank n factorization of $\mathbf{H}_f$. This construction is the cornerstone of the spectral learning algorithm and is given in the following corollary.

Corollary 2. *[3, Lemma 4.1] Let $f : \Sigma^* \to \mathbb{R}$ be a recognizable function with rank n, let $\mathbf{H} \in \mathbb{R}^{\Sigma^*\times\Sigma^*}$ be its Hankel matrix, and for each $\sigma \in \Sigma$ let $\mathbf{H}^\sigma \in \mathbb{R}^{\Sigma^*\times\Sigma^*}$ be defined by $\mathbf{H}^\sigma_{u,v} = f(u\sigma v)$ for all $u, v \in \Sigma^*$.*

Then, for any $\mathbf{P} \in \mathbb{R}^{\Sigma^*\times n}$, $\mathbf{S} \in \mathbb{R}^{n\times\Sigma^*}$ such that $\mathbf{H} = \mathbf{PS}$, the WFA $A = (\boldsymbol{\alpha}, \{\mathbf{A}^\sigma\}_{\sigma\in\Sigma}, \boldsymbol{\omega})$ where $\boldsymbol{\alpha}^\top = \mathbf{P}_{\lambda,:}$, $\boldsymbol{\omega} = \mathbf{S}_{:,\lambda}$, and $\mathbf{A}^\sigma = \mathbf{P}^\dagger\mathbf{H}^\sigma\mathbf{S}^\dagger$ is a minimal WFA for f.

In practice, finite sub-blocks of the Hankel matrices are used. Given finite sets of prefixes and suffixes $\mathcal{P}, \mathcal{S} \subset \Sigma^*$, let $\mathbf{H}_{\mathcal{P},\mathcal{S}}, \{\mathbf{H}^\sigma_{\mathcal{P},\mathcal{S}}\}_{\sigma\in\Sigma}$ be the finite sub-blocks of $\mathbf{H}$ whose rows (resp. columns) are indexed by prefixes in $\mathcal{P}$ (resp. suffixes in $\mathcal{S}$). One can show that if $\mathcal{P}$ and $\mathcal{S}$ are such that $\lambda \in \mathcal{P}\cap\mathcal{S}$ and $\operatorname{rank}(\mathbf{H}) = \operatorname{rank}(\mathbf{H}_{\mathcal{P},\mathcal{S}})$, then the previous corollary still holds, i.e. a minimal WFA computing f can be recovered from any rank n factorization of $\mathbf{H}_{\mathcal{P},\mathcal{S}}$. The spectral method thus consists in estimating the matrices $\mathbf{H}_{\mathcal{P},\mathcal{S}}, \mathbf{H}^\sigma_{\mathcal{P},\mathcal{S}}$ from training data (using e.g. empirical frequencies if f is stochastic), finding a low-rank factorization of $\mathbf{H}_{\mathcal{P},\mathcal{S}}$ (using e.g. SVD) and constructing a WFA approximating f using Corollary 2.

Tensors. We make a sporadic use of tensors in this paper, we thus introduce the few necessary definitions and notations; more details can be found in [15]. A *3rd order tensor* $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{d_1\times d_2\times d_3}$ can be seen as a multidimensional array $(\boldsymbol{\mathcal{T}}_{i_1,i_2,i_3} \ : i_1 \in [d_1], i_2 \in [d_2], , i_3 \in [d_3])$. The *mode-$n$* fibers of $\boldsymbol{\mathcal{T}}$ are the vectors obtained by fixing all indices except the nth one, e.g. $\boldsymbol{\mathcal{T}}_{:,i_2,i_3} \in \mathbb{R}^{d_1}$. The *$n$th mode flattening* of $\boldsymbol{\mathcal{T}}$ is the matrix having the mode-n fibers of $\boldsymbol{\mathcal{T}}$ for columns and is denoted by e.g. $\boldsymbol{\mathcal{T}}_{(1)} \in \mathbb{R}^{d_1\times d_2 d_3}$. The *mode-1 matrix product* of a tensor $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{d_1\times d_2\times d_3}$ and a matrix $\mathbf{X} \in \mathbb{R}^{m\times d_1}$ is a tensor of size $m \times d_2 \times d_3$ denoted by $\boldsymbol{\mathcal{T}} \times_1 \mathbf{X}$ and defined by the relation $\boldsymbol{\mathcal{Y}} = \boldsymbol{\mathcal{T}} \times_1 \mathbf{X} \Leftrightarrow \boldsymbol{\mathcal{Y}}_{(1)} = \mathbf{X}\boldsymbol{\mathcal{T}}_{(1)}$; the mode-$n$ product for $n = 2, 3$ is defined similarly.

3 Vector-Valued WFAs for Multitask Learning

In this section, we present a notion of *relatedness between WFAs* that we formalize by introducing the novel model of *vector-valued weighted automaton*. We then propose a multitask learning algorithm for WFAs by designing a spectral learning algorithm for vector-valued WFAs.

A notion of relatedness between WFAs. The basic idea behind our approach emerges from interpreting the computation of a WFA as a linear model in some feature space. Indeed, the computation of a WFA $A = (\boldsymbol{\alpha}, \{\mathbf{A}^\sigma\}_{\sigma\in\Sigma}, \boldsymbol{\omega})$ with n states on a word $x \in \Sigma^*$ can be seen as first mapping x to an n-dimensional feature vector through a *compositional feature map* $\phi : \Sigma^* \to \mathbb{R}^n$, and then applying a linear form in the feature space to obtain the final value $f_A(x) = \langle\phi(x), \boldsymbol{\omega}\rangle$. The feature map is defined by $\phi(x)^\top = \boldsymbol{\alpha}^\top\mathbf{A}^x$ for all $x \in \Sigma^*$ and it is compositional in the sense that for any $x \in \Sigma^*$ and any $\sigma \in \Sigma$ we have $\phi(x\sigma)^\top = \phi(x)^\top\mathbf{A}^\sigma$. We will say that such a feature map is *minimal* if the linear space $V \subset \mathbb{R}^n$ spanned by the vectors $\{\phi(x)\}_{x\in\Sigma^*}$ is of dimension n. Theorem 1 implies that the dimension of V is actually equal to the rank of f_A, showing that the notion of minimal feature map naturally coincides with the notion of minimal WFA.

A notion of *relatedness between WFAs* naturally arises by considering to which extent two (or more) WFAs can share a joint feature map ϕ. More precisely, consider two recognizable functions $f_1, f_2 : \Sigma^* \to \mathbb{R}$ of rank n_1 and n_2 respectively, with corresponding feature maps $\phi_1 : \Sigma^* \to \mathbb{R}^{n_1}$ and $\phi_2 : \Sigma^* \to \mathbb{R}^{n_2}$. Then, a joint feature map for f_1 and f_2 always exists and is obtained by considering the direct sum $\phi_1 \oplus \phi_2 : \Sigma^* \to \mathbb{R}^{n_1+n_2}$ that simply concatenates the feature vectors $\phi_1(x)$ and $\phi_2(x)$ for any $x \in \Sigma^*$. However, this feature map may not be minimal, i.e. there may exist another joint feature map of dimension $n < n_1 + n_2$. Intuitively, the smaller this minimal dimension n is the more related the two tasks are, with the two extremes being on the one hand $n = n_1 + n_2$ where the two tasks are independent, and on the other hand e.g. $n = n_1$ where one of the (minimal) feature maps ϕ_1, ϕ_2 is sufficient to predict both tasks.

Vector-valued WFA. We now introduce a computational model for vector-valued functions on strings that will help formalize this notion of relatedness between WFAs.

Definition 1. *A d-dimensional* vector-valued weighted finite automaton *(vv-WFA) with n states is a tuple $A = (\boldsymbol{\alpha}, \{\mathbf{A}^\sigma\}_{\sigma\in\Sigma}, \boldsymbol{\Omega})$ where $\boldsymbol{\alpha} \in \mathbb{R}^n$ is the initial weights vector, $\boldsymbol{\Omega} \in \mathbb{R}^{n\times d}$ is the matrix of final weights, and $\mathbf{A}^\sigma \in \mathbb{R}^{n\times n}$ is the transition matrix for each symbol $\sigma \in \Sigma$. A vv-WFA computes a function $\vec{f}_A : \Sigma^* \to \mathbb{R}^d$ defined by*

$$\vec{f}_A(x) = \boldsymbol{\alpha}^\top \mathbf{A}^{x_1}\mathbf{A}^{x_2}\cdots\mathbf{A}^{x_k}\boldsymbol{\Omega}$$

for each word $x = x_1x_2\cdots x_k \in \Sigma^*$.

We extend the notions of recognizability, minimality and rank of a WFA in the straightforward way: a function $\vec{f} : \Sigma^* \to \mathbb{R}^d$ is recognizable if it can be computed by a vv-WFA, a vv-WFA is minimal if its number of states is minimal, and the rank of $\vec{f}$ is the number of states of a minimal vv-WFA computing $\vec{f}$. A d-dimensional vv-WFA can be seen as a collection of d WFAs that all share their initial vectors and transition matrices but have different final vectors. Alternatively, one could take a dual approach and define vv-WFAs as a collection of WFAs sharing transitions and final vectors[1].

vv-WFAs and relatedness between WFAs. We now show how the vv-WFA model naturally captures the notion of relatedness presented above. Recall that this notion intends to capture to which extent two recognizable functions $f_1, f_2 : \Sigma^* \to \mathbb{R}$, of ranks n_1 and n_2 respectively, can share a joint forward feature map $\phi : \Sigma^* \to \mathbb{R}^n$ satisfying $f_1(x) = \langle\phi(x), \boldsymbol{\omega}_1\rangle$ and $f_2(x) = \langle\phi(x), \boldsymbol{\omega}_2\rangle$ for all $x \in \Sigma^*$, for some $\boldsymbol{\omega}_1, \boldsymbol{\omega}_2 \in \mathbb{R}^n$. Consider the vector-valued function $\vec{f} = [f_1, f_2] : \Sigma^* \to \mathbb{R}^2$ defined by $\vec{f}(x) = [f_1(x), f_2(x)]$ for all $x \in \Sigma^*$. It can easily be seen that the minimal dimension of a shared forward feature map between f_1 and f_2 is exactly the rank of $\vec{f}$, i.e. the number of states of a minimal vv-WFA computing $\vec{f}$. This notion of relatedness can be generalized to more than two functions by considering $\vec{f} = [f_1, \cdots, f_m]$ for m different recognizable functions $f_1, \cdots, f_m$ of respective ranks $n_1, \cdots, n_m$. In this setting, it is easy to check that the rank of $\vec{f}$ lies between $\max(n_1, \cdots, n_m)$ and $n_1 + \cdots + n_m$; smaller values of this rank leads to a smaller dimension of the minimal forward feature map and thus, intuitively, to more closely related tasks. We now formalize this measure of relatedness between recognizable functions.

Definition 2. *Given m recognizable functions $f_1, \cdots, f_m$, we define their* relatedness measure *by $\tau(f_1, \cdots, f_m) = 1 - (\mathrm{rank}(\vec{f}) - \max_i \mathrm{rank}(f_i))/\sum_i \mathrm{rank}(f_i)$ where $\vec{f} = [f_1, \cdots, f_m]$.*

One can check that this measure of relatedness takes its values in $(0, 1]$. We say that tasks are *maximally related* when their relatedness measure is 1 and *independent* when it is minimal.

Example 1. *Let $\Sigma = \{a, b, c\}$ and let $|x|_\sigma$ denotes the number of occurrences of σ in x for any $\sigma \in \Sigma$. Consider the functions defined by $f_1(x) = 0.5|x|_a + 0.5|x|_b$, $f_2(x) = 0.3|x|_b - 0.6|x|_c$ and $f_3(x) = |x|_c$ for all $x \in \Sigma^*$. It is easy to check that $\mathrm{rank}(f_1) = \mathrm{rank}(f_2) = 4$ and $\mathrm{rank}(f_3) = 2$. Moreover, f_2 and f_3 are maximally related (indeed $\mathrm{rank}([f_2, f_3]) = 4 = \mathrm{rank}(f_2)$ thus $\tau(f_2, f_3) = 1$), f_1 and f_3 are independent (indeed $\tau(f_1, f_3) = 2/3$ is minimal since $\mathrm{rank}([f_1, f_3]) = 6 = \mathrm{rank}(f_1) + \mathrm{rank}(f_3)$), and f_1 and f_2 are related but not maximally related (since $4 = \mathrm{rank}(f_1) = \mathrm{rank}(f_2) < \mathrm{rank}([f_1, f_2]) = 6 < \mathrm{rank}(f_1) + \mathrm{rank}(f_2) = 8$).*

[1] Both definitions performed similarly in multitask experiments on the dataset used in Section 5.2, we thus chose multiple final vectors as a convention.

Spectral learning of vv-WFAs. We now design a spectral learning algorithm for vv-WFAs. Given a function $\vec{f}:\Sigma^*\to\mathbb{R}^d$, we define its Hankel tensor $\boldsymbol{\mathcal{H}}\in\mathbb{R}^{\Sigma^*\times d\times\Sigma^*}$ by $\boldsymbol{\mathcal{H}}_{u,:,v}=\vec{f}(uv)$ for all $u,v\in\Sigma^*$. We first show in Theorem 3 (whose proof can be found in the appendix) that the fundamental relation between the rank of a function and the rank of its Hankel matrix can naturally be extended to the vector-valued case. Compared with Theorem 1, the Hankel matrix is now replaced by the mode-1 flattening $\boldsymbol{\mathcal{H}}_{(1)}$ of the Hankel tensor (which can be obtained by concatenating the matrices $\boldsymbol{\mathcal{H}}_{:,i,:}$ along the horizontal axis).

Theorem 3 (Vector-valued Fliess Theorem)**.** *Let $\vec{f}:\Sigma^*\to\mathbb{R}^d$ and let $\boldsymbol{\mathcal{H}}$ be its Hankel tensor. Then* $\mathrm{rank}(\vec{f})=\mathrm{rank}(\boldsymbol{\mathcal{H}}_{(1)})$.

Similarly to the scalar-valued case, this theorem can be leveraged to design a spectral learning algorithm for vv-WFAs. The following corollary (whose proof can be found in the appendix) shows how a vv-WFA computing a recognizable function $\vec{f}:\Sigma^*\to\mathbb{R}^d$ of rank n can be recovered from any rank n factorization of its Hankel tensor.

Corollary 4. *Let $\vec{f}:\Sigma^*\to\mathbb{R}^d$ be a recognizable function with rank n, let $\boldsymbol{\mathcal{H}}\in\mathbb{R}^{\Sigma^*\times d\times\Sigma^*}$ be its Hankel tensor, and for each $\sigma\in\Sigma$ let $\boldsymbol{\mathcal{H}}^\sigma\in\mathbb{R}^{\Sigma^*\times d\times\Sigma^*}$ be defined by $\boldsymbol{\mathcal{H}}^\sigma_{u,:,v}=\vec{f}(u\sigma v)$ for all $u,v\in\Sigma^*$.*

Then, for any $\mathbf{P}\in\mathbb{R}^{\Sigma^*\times n}$ and $\boldsymbol{\mathcal{S}}\in\mathbb{R}^{n\times d\times\Sigma^*}$ such that $\boldsymbol{\mathcal{H}}=\boldsymbol{\mathcal{S}}\times_1\mathbf{P}$, the vv-WFA $A=(\boldsymbol{\alpha},\{\mathbf{A}^\sigma\}_{\sigma\in\Sigma},\boldsymbol{\Omega})$ defined by $\boldsymbol{\alpha}^\top=\mathbf{P}_{\lambda,:}$, $\boldsymbol{\Omega}=\boldsymbol{\mathcal{S}}_{:,:,\lambda}$, and $\mathbf{A}^\sigma=\mathbf{P}^\dagger\boldsymbol{\mathcal{H}}^\sigma_{(1)}(\boldsymbol{\mathcal{S}}_{(1)})^\dagger$ is a minimal vv-WFA computing $\vec{f}$.

Similarly to the scalar-valued case, one can check that the previous corollary also holds for any finite sub-tensors $\boldsymbol{\mathcal{H}}_{\mathcal{P},\mathcal{S}},\{\boldsymbol{\mathcal{H}}^\sigma_{\mathcal{P},\mathcal{S}}\}_{\sigma\in\Sigma}$ of $\boldsymbol{\mathcal{H}}$ indexed by prefixes and suffixes in $\mathcal{P},\mathcal{S}\subset\Sigma^*$, whenever $\mathcal{P}$ and $\mathcal{S}$ are such that $\lambda\in\mathcal{P}\cap\mathcal{S}$ and $\mathrm{rank}(\boldsymbol{\mathcal{H}}_{(1)})=\mathrm{rank}((\boldsymbol{\mathcal{H}}_{\mathcal{P},\mathcal{S}})_{(1)})$; we will call such a basis $(\mathcal{P},\mathcal{S})$ *complete*. The spectral learning algorithm for vv-WFAs then consists in estimating these Hankel tensors from training data and using Corollary 4 to recover a vv-WFA approximating the target function. Of course a noisy estimate of the Hankel tensor $\hat{\boldsymbol{\mathcal{H}}}$ will not be of low rank and the factorization $\hat{\boldsymbol{\mathcal{H}}}=\boldsymbol{\mathcal{S}}\times_1\mathbf{P}$ should only be performed approximately in order to counter the presence of noise. In practice a low rank approximation of $\hat{\boldsymbol{\mathcal{H}}}_{(1)}$ is obtained using truncated SVD.

Multitask learning of WFAs. Let us now go back to the multitask learning problem and let $f_1,\cdots f_m:\Sigma^*\to\mathbb{R}$ be multiple functions we wish to infer in the form of WFAs. The spectral learning algorithm for vv-WFAs naturally suggests a way to tackle this multitask problem: by learning $\vec{f}=[f_1,\cdots,f_m]$ in the form of a vv-WFA, rather than independently learning a WFA for each task f_i, we implicitly enforce the discovery of a joint forward feature map shared among all tasks.

We will now see how a further step can be added to this learning scheme to enforce more robustness to noise. The motivation for this additional step comes from the observation that even though a d-dimensional vv-WFA $A=(\boldsymbol{\alpha},\{\mathbf{A}^\sigma\}_{\sigma\in\Sigma},\boldsymbol{\Omega})$ may be minimal, the corresponding scalar-valued WFAs $A_i=\langle\boldsymbol{\alpha},\{\mathbf{A}^\sigma\}_{\sigma\in\Sigma},\boldsymbol{\Omega}_{:,i}\rangle$ for $i\in[d]$ may not be. Suppose for example that A_1 is not minimal. This implies that some part of its state space does not contribute to the function f_1 but comes from asking for a rich enough state representation that can predict other tasks as well. Moreover, when one learns a vv-WFA from noisy estimates of the Hankel tensors, the rank R approximation $\hat{\boldsymbol{\mathcal{H}}}_{(1)}\simeq\mathbf{P}\boldsymbol{\mathcal{S}}_{(1)}$ somehow annihilates the noise contained in the space orthogonal to the top R singular vectors of $\hat{\boldsymbol{\mathcal{H}}}_{(1)}$, but when the WFA A_1 has rank $R_1<R$ we intuitively see that there is still a subspace of dimension $R-R_1$ containing only irrelevant features. In order to circumvent this issue, we would like to project down the (scalar-valued) WFAs A_i down to their true dimensions, intuitively enforcing each predictor to use as few features as possible for each task, and thus annihilating the noise lying in the corresponding irrelevant subspaces. To achieve this we will make use of the following proposition that explicits the projections needed to obtain minimal scalar-valued WFAs from a given vv-WFA (the proof is given in the appendix).

Proposition 1. *Let $\vec{f}:\Sigma^*\to\mathbb{R}^d$ be a function computed by a minimal vv-WFA $A=(\boldsymbol{\alpha},\{\mathbf{A}^\sigma\}_{\sigma\in\Sigma},\boldsymbol{\Omega})$ with n states and let $\mathcal{P},\mathcal{S}\subseteq\Sigma^*$ be a complete basis for $\vec{f}$. For any $i\in[d]$, let $f_i:\Sigma^*\to\mathbb{R}$ be defined by $f_i(x)=\vec{f}(x)_i$ for all $x\in\Sigma^*$ and let n_i denote the rank of f_i.*

Let $\mathbf{P}\in\mathbb{R}^{\mathcal{P}\times n}$ be defined by $\mathbf{P}_{x,:}=\boldsymbol{\alpha}^\top\mathbf{A}^x$ for all $x\in\mathcal{P}$ and, for $i\in[d]$, let $\mathbf{H}_i\in\mathbb{R}^{\mathcal{P}\times\mathcal{S}}$ be the Hankel matrix of f_i and let $\mathbf{H}_i=\mathbf{U}_i\mathbf{D}_i\mathbf{V}_i^\top$ be its thin SVD (i.e. $\mathbf{D}_i\in\mathbb{R}^{n_i\times n_i}$).

Then, for any $i\in[d]$, the WFA $A_i=\langle\boldsymbol{\alpha}_i,\{\mathbf{A}_i^\sigma\}_{\sigma\in\Sigma}\},\boldsymbol{\omega}_i\rangle$ defined by

$$\boldsymbol{\alpha}_i^\top=\boldsymbol{\alpha}^\top\mathbf{P}^\dagger\mathbf{U}_i,\ \boldsymbol{\omega}_i=\mathbf{U}_i^\top\mathbf{P}\boldsymbol{\Omega}_{:,i}\ \text{and}\ \mathbf{A}_i^\sigma=\mathbf{U}_i^\top\mathbf{P}\mathbf{A}^\sigma\mathbf{P}^\dagger\mathbf{U}_i\ \text{for each}\ \sigma\in\Sigma,$$

is a minimal WFA computing f_i.

Given noisy estimates $\hat{\boldsymbol{\mathcal{H}}},\{\hat{\boldsymbol{\mathcal{H}}}^\sigma\}_{\sigma\in\Sigma}$ of the Hankel tensors of a function $\vec{f}$ and estimates R of the rank of $\vec{f}$ and R_i of the ranks of the f_i's, the first step of the learning algorithm consists in applying Corollary 4 to the factorization $\hat{\boldsymbol{\mathcal{H}}}_{(1)}\simeq\mathbf{U}(\mathbf{D}\mathbf{V}^\top)$ obtained by truncated SVD to get a vv-WFA A approximating $\vec{f}$. Then, Proposition 1 can be used to project down each WFA A_i by estimating $\mathbf{U}_i$ with the top R_i left singular vectors of $\hat{\boldsymbol{\mathcal{H}}}_{:,i,:}$. The overall procedure for our Multi-Task Spectral Learning (MT-SL) is summarized in Algorithm 1 where lines 1-3 correspond to the vv-WFA estimation while lines 4-7 correspond to projecting down the corresponding scalar-valued WFAs.

Algorithm 1 `MT-SL`: Spectral Learning of vector-valued WFA for multitask learning

---

Input: Empirical Hankel tensors $\hat{\boldsymbol{\mathcal{H}}},\{\hat{\boldsymbol{\mathcal{H}}}^\sigma\}_{\sigma\in\Sigma}$ of size $\mathcal{P}\times m\times\mathcal{S}$ for the target function $\vec{f}=[f_1,\cdots,f_m]$ (where $\mathcal{P},\mathcal{S}$ are subsets of Σ^* both containing λ), a common rank R, and task specific ranks R_i for $i\in[m]$.

Output: WFAs A_i approximating f_i for each $i\in[d]$.

1: Compute the rank R truncated SVD $\hat{\boldsymbol{\mathcal{H}}}_{(1)}\simeq\mathbf{U}\mathbf{D}\mathbf{V}^\top$.

2: Let $A=(\boldsymbol{\alpha},\{\mathbf{A}^\sigma\}_{\sigma\in\Sigma},\boldsymbol{\Omega})$ be the vv-WFA defined by
$$\boldsymbol{\alpha}^\top=\mathbf{U}_{\lambda,:},\quad,\boldsymbol{\Omega}=\mathbf{U}^\top(\hat{\boldsymbol{\mathcal{H}}}_{:,:,\lambda})\quad\text{and}\quad\mathbf{A}^\sigma=\mathbf{U}^\top\hat{\boldsymbol{\mathcal{H}}}^\sigma_{(1)}(\hat{\boldsymbol{\mathcal{H}}}_{(1)})^\dagger\mathbf{U}\ \text{ for each }\sigma\in\Sigma.$$

3: **for** $i=1$ **to** m **do**

4: Compute the rank R_i truncated SVD $\hat{\boldsymbol{\mathcal{H}}}_{:,i,:}\simeq\mathbf{U}_i\mathbf{D}_i\mathbf{V}_i^\top$.

5: Let $A_i=\langle\mathbf{U}_i^\top\mathbf{U}\boldsymbol{\alpha},\{\mathbf{U}_i^\top\mathbf{U}\mathbf{A}^\sigma\mathbf{U}^\top\mathbf{U}_i\}_{\sigma\in\Sigma},\mathbf{U}_i^\top\mathbf{U}\boldsymbol{\Omega}_{:,i}\rangle$

6: **end for**

7: **return** $A_1,\cdots,A_m$.

---

4 Theoretical Analysis

Computational complexity. The computational cost of the classical spectral learning algorithm (SL) is in $\mathcal{O}\left(N+R|\mathcal{P}||\mathcal{S}|+R^2|\mathcal{P}||\Sigma|\right)$ where the first term corresponds to estimating the Hankel matrices from a sample of size N, the second one to the rank R truncated SVD, and the third one to computing the transition matrices $\mathbf{A}^\sigma$. In comparison, the computational cost of MT-SL is in $\mathcal{O}\left(mN+(mR+\sum_i R_i)|\mathcal{P}||\mathcal{S}|+(mR^2+\sum_i R_i^2)|\mathcal{P}||\Sigma|\right)$, showing that the increase in complexity is essentially linear in the number of tasks m.

Robustness in subspace estimation. In order to give some theoretical insights on the potential benefits of MT-SL, let us consider the simple case where the tasks are maximally related with common rank $R=R_1=\cdots=R_m$. Let $\hat{\mathbf{H}}_1,\cdots,\hat{\mathbf{H}}_m\in\mathbb{R}^{\mathcal{P}\times\mathcal{S}}$ be the empirical Hankel matrices for the m tasks and let $\mathbf{E}_i=\hat{\mathbf{H}}_i-\mathbf{H}_i$ be the error terms, where $\mathbf{H}_i$ is the true Hankel matrix for the ith task. Then the flattening $\hat{\mathbf{H}}=\hat{\boldsymbol{\mathcal{H}}}_{(1)}\in\mathbb{R}^{|\mathcal{P}|\times m|\mathcal{S}|}$ (resp. $\mathbf{H}=\boldsymbol{\mathcal{H}}_{(1)}$) can be obtained by stacking the matrices $\hat{\mathbf{H}}_i$ (resp. $\mathbf{H}_i$) along the horizontal axis. Consider the problem of learning the first task. One key step of both SL and MT-SL resides in estimating the left singular subspace of $\mathbf{H}_1$ and $\mathbf{H}$ respectively from their noisy estimates. When the tasks are maximally related, this space $\mathcal{U}$ is the same for $\mathbf{H}$ and $\mathbf{H}_1,\cdots,\mathbf{H}_m$

and we intuitively see that the benefits of MT-SL will stem from the fact that the SVD of $\hat{\mathbf{H}}$ should lead to a more accurate estimation of $\mathcal{U}$ than the one only relying on $\hat{\mathbf{H}}_1$. It is also intuitive to see that since the Hankel matrices $\hat{\mathbf{H}}_i$ have been stacked horizontally, the estimation of the right singular subspace might not benefit from performing SVD on $\hat{\mathbf{H}}$. However, classical results on singular subspace estimation (see e.g. [25, 16]) provide uniform bounds for both left and right singular subspaces (i.e. bounds on the maximum of the estimation errors for the left and right spaces). To circumvent this issue, we use a recent result on rate optimal asymmetric perturbation bounds for left and right singular spaces [6] to obtain the following theorem relating the ratio between the dimensions of a matrix to the quality of the subspace estimation provided by SVD (the proof can be found in the appendix).

Theorem 5. *Let $\mathbf{M} \in \mathbb{R}^{d_1 \times d_2}$ be of rank R and let $\hat{\mathbf{M}} = \mathbf{M} + \mathbf{E}$ where $\mathbf{E}$ is a random noise term such that $\frac{\mathrm{vec}(\mathbf{E})}{\|\mathbf{E}\|_F}$ follows a uniform distribution on the unit sphere in $\mathbb{R}^{d_1 d_2}$. Let $\mathbf{\Pi}_U, \mathbf{\Pi}_{\hat{U}} \in \mathbb{R}^{d_1 \times d_1}$ be the matrices of the orthogonal projections onto the space spanned by the top R left singular vectors of $\mathbf{M}$ and $\hat{\mathbf{M}}$ respectively.*

Let $\delta > 0$, let $\alpha = \mathfrak{s}_R(\mathbf{M})$ be the smallest non-zero singular value of $\mathbf{M}$ and suppose that $\|\mathbf{E}\|_F \leq \alpha/2$. Then, with probability at least $1 - \delta$,

$$\|\mathbf{\Pi}_U - \mathbf{\Pi}_{\hat{U}}\|_F \leq 4 \left(\sqrt{\frac{(d_1 - R)R + 2\log(1/\delta)}{d_1 d_2}} \frac{\|\mathbf{E}\|_F}{\alpha} + \frac{\|\mathbf{E}\|_F^2}{\alpha^2} \right).$$

A few remarks on this theorem are in order. First, the Frobenius norm between the projection matrices measures the distance between the two subspaces (it is in fact proportional to the classical sin-theta distance between subspaces). Second, the assumption $\|\mathbf{E}\|_F \leq \alpha/2$ corresponds to the magnitude of the noise being small compared to the magnitude of $\mathbf{M}$ (and in particular it implies $\frac{\|\mathbf{E}\|_F}{\alpha} < 1$). Lastly, as d_2 grows the first term in the upper bound becomes irrelevant and the error is dominated by the quadratic term, which decreases with $\|\mathbf{E}\|_F$ faster than classical results. Intuitively this tells us that there is a first regime where growing d_2 (i.e. adding more tasks) is beneficial, until the point where the quadratic term dominates (and where the bound becomes somehow independent of d_2).

Going back to the power of MT-SL to leverage information from related tasks, let $\mathbf{E} \in \mathbb{R}^{|\mathcal{P}| \times m|\mathcal{S}|}$ be the matrix obtained by stacking the noise matrices $\mathbf{E}_i$ along the horizontal axis. If we assume that the entries of the error terms $\mathbf{E}_i$ are i.i.d. from e.g. a normal distribution, we can apply the previous proposition to the left singular subspaces of $\hat{\boldsymbol{\mathcal{H}}}_{(1)}$ and $\boldsymbol{\mathcal{H}}_{(1)}$. One can check that in this case we have $\|\mathbf{E}\|_F^2 = \sum_{i=1}^m \|\mathbf{E}_i\|_F^2$ and $\alpha^2 = \mathfrak{s}_R(\mathbf{H})^2 \geq \sum_{i=1}^m \mathfrak{s}_R(\mathbf{H}_i)^2$ (since $R = R_1 = \cdots = R_m$ when tasks are maximally related). Thus, if the norms of the noise terms $\mathbf{E}_i$ are roughly the same, and so are the smallest non-zero singular values of the matrices $\mathbf{H}_i$, we get $\frac{\|\mathbf{E}\|_F}{\alpha} \leq \mathcal{O}\left(\|\mathbf{E}_1\|_F / \mathfrak{s}_R(\mathbf{H}_1)\right)$. Hence, given enough tasks, the estimation error of the left singular subspace of $\mathbf{H}_1$ in the multitask setting (i.e. by performing SVD on $\hat{\boldsymbol{\mathcal{H}}}_{(1)}$) is intuitively in $\mathcal{O}\left(\|\mathbf{E}_1\|_F^2 / \mathfrak{s}_R(\mathbf{H}_1)^2\right)$ while it is only in $\mathcal{O}\left(\|\mathbf{E}_1\|_F / \mathfrak{s}_R(\mathbf{H}_1)\right)$ when relying solely on $\hat{\mathbf{H}}_1$, which shows the potential benefits of MT-SL.

5 Experiments

We evaluate the performance of the proposed multitask learning method (MT-SL) on both synthetic and real world data. We use two performance metrics: perplexity per character on a test set T, which is defined by $\mathrm{perp}(h) = 2^{-\frac{1}{M}\sum_{x \in T} \log(h(x))}$ where M is the number of symbols in the test set and h is the hypothesis, and word error rate (WER) which measures the proportion of mis-predicted symbols averaged over all prefixes in the test set (when the most likely symbol is predicted). Both experiments are in a stochastic setting, i.e. the functions to be learned are probability distributions, and explore the regime where the learner has access to a small training sample drawn from the target task, while larger training samples are available for related tasks. We compare MT-SL with the classical spectral learning method (SL) for WFAs [3]. For both methods the prefix set $\mathcal{P}$ (resp. suffix set $\mathcal{S}$) is chosen by taking

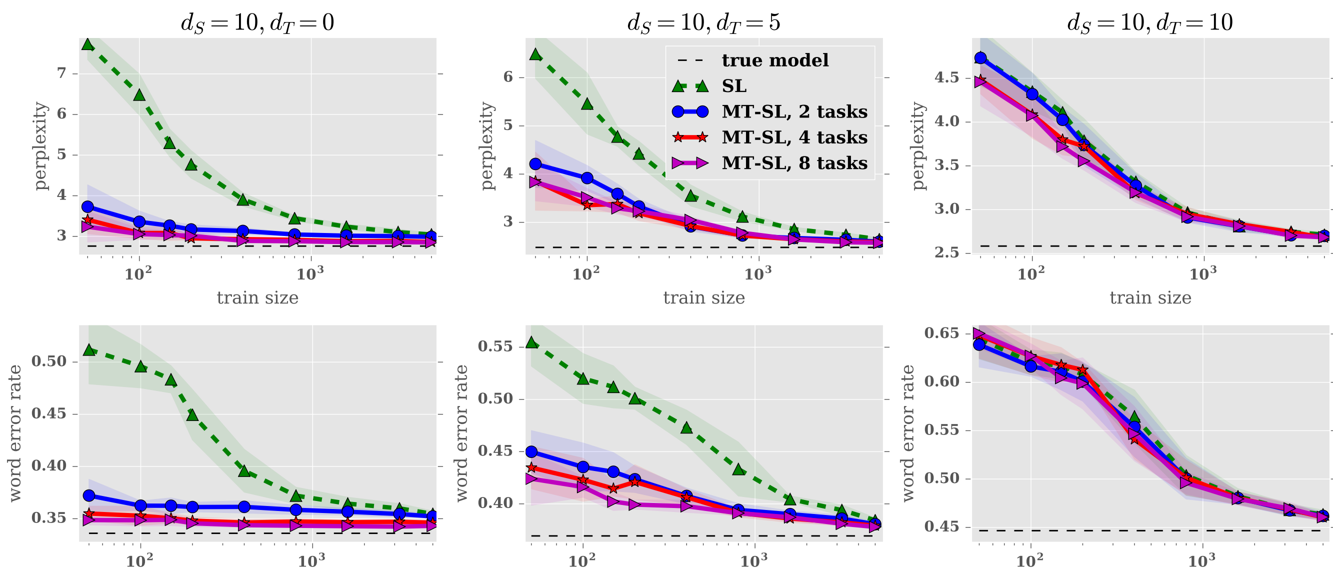

Figure 1: Comparison (on synthetic data) between the spectral learning algorithm (SL) and our multitask algorithm (MT-SL) for different numbers of tasks and different degrees of relatedness between the tasks: d_S is the dimension of the space shared by all tasks and d_T the one of the task-specific space (see text for details).

the $1,000$ most frequent prefixes (resp. suffixes) in the training data of the target task, and the values of the ranks are chosen using a validation set.

5.1 Synthetic Data

We first assess the validity of MT-SL on synthetic data. We randomly generated stochastic WFAs using the process used for the PAutomaC competition [23] with symbol sparsity 0.4 and transition sparsity 0.15, for an alphabet Σ of size 10. We generated related WFAs[2] sharing a joint feature space of dimension $d_S = 10$ and each having a task specific feature space of dimension d_T, i.e. for m tasks $f_1, \cdots, f_m$ each WFA computing f_i has rank $d_S + d_T$ and the vv-WFA computing $\vec{f} = [f_1, \cdots, f_m]$ has rank $d_S + md_T$. We generated 3 sets of WFAs for different task specific dimensions $d_T = 0, 5, 10$. The learner had access to training samples of size $5,000$ drawn from each related tasks $f_2, \cdots, f_m$ and a training sample of sizes ranging from 50 to $5,000$ drawn from the target task f_1. Results on a test set of size $1,000$ averaged over 10 runs are reported in Figure 1.

For both evaluation measures, when the task specific dimension is small compared to the dimension of the joint feature space, i.e. $d_T = 0, 5$, MT-SL clearly outperforms SL that only relies on the target task data. Moreover, increasing the number of related tasks tends to improve the performances of MT-SL. However, when $d_S = d_T = 10$, MT-SL performs similarly in terms of perplexity and WER, showing that the multitask approach offers no benefits when the tasks are too loosely related.

5.2 Real Data

We evaluate MT-SL on 33 languages from the Universal Dependencies (UNIDEP) 1.4 treebank [20], using the 17-tag universal Part of Speech (PoS) tagset. This dataset contains sentences from various

[2] More precisely, we first generate a probabilistic automaton (PA) $A_S = (\boldsymbol{\alpha}_S, \{\mathbf{A}_S^\sigma\}_{\sigma\in\Sigma}, \boldsymbol{\omega}_S)$ with d_S states. Then, for each task $i = 1, \cdots, m$ we generate a second PA $A_T = (\boldsymbol{\alpha}_T, \{\mathbf{A}_T^\sigma\}_{\sigma\in\Sigma}, \boldsymbol{\omega}_T)$ with d_T states and a random vector $\boldsymbol{\omega} \in [0,1]^{d_S+d_T}$. Both PAs are generated using the process described in [23]. The task f_i is then obtained as the distribution computed by the stochastic WFA $\langle \boldsymbol{\alpha}_S \oplus \boldsymbol{\alpha}_T, \{\mathbf{A}_S^\sigma \oplus \mathbf{A}_T^\sigma\}_{\sigma\in\Sigma}, \tilde{\boldsymbol{\omega}}\rangle$ with $\tilde{\boldsymbol{\omega}} = \boldsymbol{\omega}/Z$ where the constant Z is chosen such that $\sum_{x\in\Sigma^*} f_i(x) = 1$.

Training size	100	500	1000	5000	all available data
	Related tasks: all other languages				
Perplexity	6.0811 ($\pm$7.82)	3.4462 ($\pm$5.32)	2.9733 ($\pm$5.23)	3.5610 ($\pm$5.30)	3.1141 ($\pm$5.42)
WER	1.2103 ($\pm$1.88)	0.8234 ($\pm$2.10)	1.1707 ($\pm$2.38)	1.7114 ($\pm$2.73)	1.5920 ($\pm$2.70)
	Related tasks: 4 closest languages				
Perplexity	6.6447 ($\pm$7.84)	4.3097 ($\pm$5.57)	3.7982 ($\pm$5.24)	3.1971 ($\pm$5.63)	2.7866 ($\pm$5.84)
WER	2.0135 ($\pm$2.81)	1.7025 ($\pm$2.71)	1.2685 ($\pm$2.10)	1.4412 ($\pm$2.06)	1.3126 ($\pm$2.24)

Table 1: Average relative improvement (in %) of the multitask approach on the UNIDEP dataset.

languages where each word is annotated with Google universal PoS tags [21], and thus can be seen as a collection of samples drawn from 33 distributions over strings on an alphabet of size 17. For each language, the available data is split between a training, a validation and a test set (80%, 10%, 10%). For each language and for various sizes of training samples, we compare independently learning the target task with SL against using MT-SL to exploit training data from related tasks. We tested two ways of selecting the related tasks: (1) all other languages are used and (2) for each language we selected the 4 closest languages w.r.t. the distance between the subspaces spanned by the top 50 left singular vectors of their Hankel matrices [3].

We report the average relative improvement of MT-SL w.r.t. SL over all languages in Table 1, e.g. for perplexity we report $100 \cdot (p_{sl} - p_{mt})/p_{sl}$ where p_{sl} (resp. p_{mt}) is the perplexity obtained by SL (resp. MT-SL) on the test set. We see that the multitask approach leads to improved results for both metrics, that the benefits tend to be greater for small training sizes, and that restricting the number of auxiliary tasks is overall beneficial. To give a concrete example, on the Basque task with a training set of size 500, the WER was reduced from $\sim 77\%$ for SL to $\sim 71\%$ using all other languages as related tasks, and to $\sim 68\%$ using the 4 closest tasks (Finnish, Polish, Czech and Indonesian). The detailed results on all languages, along with the list of closest languages used for method (2), are reported in the appendix.

6 Conclusion

We introduced the novel model of vector-valued WFA that allowed us to define a notion of relatedness between recognizable functions and to design a multitask spectral learning algorithm for WFAs (MT-SL). The benefits of MT-SL have been theoretically motivated and showcased on both synthetic and real data experiments. In future works, we plan to apply MT-SL in the context of reinforcement learning and to identify other areas of machine learning where vv-WFAs could prove to be useful.

References

[1] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Multi-task feature learning. In *NIPS*, pages 41–48, 2007.

[2] Raphaël Bailly, François Denis, and Liva Ralaivola. Grammatical inference as a principal component analysis problem. In *ICML*, pages 33–40, 2009.

[3] Borja Balle, Xavier Carreras, Franco M Luque, and Ariadna Quattoni. Spectral learning of weighted automata. *Machine learning*, 96(1-2):33–63, 2014.

[4] Jonathan Baxter et al. A model of inductive bias learning. *Journal of Artifical Intelligence Research*, 12(149-198):3, 2000.

[3] The common basis $(\mathcal{P}, \mathcal{S})$ for these Hankel matrices is chosen by taking the union of the 100 most frequent prefixes and suffixes in each training sample.

[5] Shai Ben-David and Reba Schuller. Exploiting task relatedness for multiple task learning. In *Learning Theory and Kernel Machines*, pages 567–580. Springer, 2003.

[6] T Tony Cai and Anru Zhang. Rate-optimal perturbation bounds for singular subspaces with applications to high-dimensional statistics. *arXiv preprint arXiv:1605.00353*, 2016.

[7] Jack W. Carlyle and Azaria Paz. Realizations by stochastic finite automata. *Journal of Computer and System Sciences*, 5(1):26–40, 1971.

[8] Rich Caruana. Multitask learning. In *Learning to learn*, pages 95–133. Springer, 1998.

[9] François Denis and Yann Esposito. On rational stochastic languages. *Fundamenta Informaticae*, 86(1, 2):41–77, 2008.

[10] Devdatt P Dubhashi and Alessandro Panconesi. *Concentration of measure for the analysis of randomized algorithms*. Cambridge University Press, 2009.

[11] Michel Fliess. Matrices de hankel. *Journal de Mathématiques Pures et Appliquées*, 53(9):197–222, 1974.

[12] Emily Fox, Michael I Jordan, Erik B Sudderth, and Alan S Willsky. Sharing features among dynamical systems with beta processes. In *NIPS*, pages 549–557, 2009.

[13] Mark A Girolami and Ata Kabán. Simplicial mixtures of markov chains: Distributed modelling of dynamic user profiles. In *NIPS*, volume 16, pages 9–16, 2003.

[14] Daniel J. Hsu, Sham M. Kakade, and Tong Zhang. A spectral algorithm for learning hidden markov models. In *COLT*, 2009.

[15] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[16] Ren-Cang Li. Relative perturbation theory: II. eigenspace and singular subspace variations. *SIAM Journal on Matrix Analysis and Applications*, 20(2):471–492, 1998.

[17] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. Recurrent neural network for text classification with multi-task learning. In *IJCAI*, pages 2873–2879, 2016.

[18] Minh-Thang Luong, Quoc V Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. Multi-task sequence to sequence learning. *arXiv preprint arXiv:1511.06114*, 2015.

[19] Kai Ni, Lawrence Carin, and David Dunson. Multi-task learning for sequential data via ihmms and the nested dirichlet process. In *ICML*, pages 689–696, 2007.

[20] Joakim Nivre, Zeljko Agić, Lars Ahrenberg, et al. Universal dependencies 1.4, 2016. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University.

[21] Slav Petrov, Dipanjan Das, and Ryan McDonald. A universal part-of-speech tagset. *arXiv preprint arXiv:1104.2086*, 2011.

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

[23] Sicco Verwer, Rémi Eyraud, and Colin De La Higuera. Results of the pautomac probabilistic automaton learning competition. In *ICGI*, pages 243–248, 2012.

[24] Boyu Wang, Joelle Pineau, and Borja Balle. Multitask generalized eigenvalue program. In *AAAI*, pages 2115–2121, 2016.

[25] Per-Åke Wedin. Perturbation bounds in connection with singular value decomposition. *BIT Numerical Mathematics*, 12(1):99–111, 1972.

A Proofs

A.1 Proof of Theorem 3

Theorem. *Let $\vec{f}: \Sigma^* \to \mathbb{R}^d$ and let $\boldsymbol{\mathcal{H}}$ be the corresponding Hankel tensor. Then* $\operatorname{rank}(f) = \operatorname{rank}(\boldsymbol{\mathcal{H}}_{(1)})$.

Proof. We first show that $\operatorname{rank}(\vec{f}) \geq \operatorname{rank}(\boldsymbol{\mathcal{H}}_{(1)})$. Let $A = (\boldsymbol{\alpha}, \{\mathbf{A}^\sigma\}_{\sigma\in\Sigma}, \boldsymbol{\Omega})$ be a vv-WFA with n states computing $\vec{f}$ and let $\mathbf{P} \in \mathbb{R}^{\Sigma^*\times n}$ and $\boldsymbol{\mathcal{S}} \in \mathbb{R}^{n\times d\times\Sigma^*}$ be defined by

$$\mathbf{P}_{u,:} = \boldsymbol{\alpha}^\top \mathbf{A}^u \quad \text{and} \quad \boldsymbol{\mathcal{S}}_{:,:,v} = \mathbf{A}^v \boldsymbol{\Omega}.$$

It is easy to check that $\boldsymbol{\mathcal{H}} = \boldsymbol{\mathcal{S}} \times_1 \mathbf{P}$ which implies $\boldsymbol{\mathcal{H}}_{(1)} = \mathbf{P}\boldsymbol{\mathcal{S}}_{(1)}$ and thus $\operatorname{rank}(\boldsymbol{\mathcal{H}}_{(1)}) \leq n$.

For the converse, we first define the notion of *residual functions* of $\vec{f}$: for any $x \in \Sigma^*$ the residual $\overline{x}: \Sigma^* \to \mathbb{R}^d$ is the function defined by $\overline{x}(u) = \vec{f}(xu)$ for any $u \in \Sigma^*$. Let $V = \{\overline{x} \ : \ x \in \Sigma^*\} \subset (\mathbb{R}^d)^{\Sigma^*}$ be the space of residual functions of f. Suppose that $\operatorname{rank}(\boldsymbol{\mathcal{H}}_{(1)}) = n$. Since each residual $\overline{x}$ can be identified with the row vector $(\boldsymbol{\mathcal{H}}_{(1)})_{x,:}$, the dimension of V is equal to n. Thus there exist n words $e_1, \cdots, e_n \in \Sigma^*$ such that $(\overline{e_1}, \cdots, \overline{e_n})$ is a basis of V. Expressing $\overline{\lambda}$ and $\overline{e_i\sigma}$ for each $i \in [n]$, $\sigma \in \Sigma$ in this basis, we know that there exist $\boldsymbol{\alpha} \in \mathbb{R}^n$ and $\mathbf{A}^\sigma \in \mathbb{R}^{n\times n}$ for each σ such that

$$\overline{\lambda} = \sum_i \boldsymbol{\alpha}_i \overline{e_i} \quad \text{and} \quad \overline{e_i\sigma} = \sum_j \mathbf{A}^\sigma_{i,j} \overline{e_j}.$$

We now show by induction on $|x|$ that $\overline{e_i x} = \sum_j \mathbf{A}^x_{i,j} \overline{e_j}$ for any non-empty string $x \in \Sigma^*$. The case $x = \sigma \in \Sigma$ is immediate by definition of $\mathbf{A}^\sigma$. Let x, y be two non-empty words, for any $u \in \Sigma^*$ and any $i \in [n]$ we get

$$\begin{aligned}
&\overline{e_i xy}(u) = \vec{f}(e_i xyu) = \overline{e_i x}(yu) \\
&= \sum_j \mathbf{A}^x_{i,j} \overline{e_j}(yu) = \sum_j \mathbf{A}^x_{i,j} \vec{f}(e_j yu) = \sum_j \mathbf{A}^x_{i,j} \overline{e_j y}(u) \\
&= \sum_j \mathbf{A}^x_{i,j} \sum_k \mathbf{A}^y_{j,k} \overline{e_k}(u) = \sum_k \mathbf{A}^{xy}_{i,k} \overline{e_k}(u)
\end{aligned}$$

using the induction hypothesis twice. To conclude the proof, let $\boldsymbol{\Omega} \in \mathbb{R}^{n\times d}$ be the matrix with rows $\overline{e_i}(\lambda)$ for $i \in [n]$. For any $x \in \Sigma^*$ we have

$$\begin{aligned}
\vec{f}(x) = \overline{\lambda}(x) &= \sum_i \boldsymbol{\alpha}_i \overline{e_i}(x) = \sum_i \boldsymbol{\alpha}_i \overline{e_i x}(\lambda) \\
&= \sum_i \boldsymbol{\alpha}_i \sum_j \mathbf{A}^x_{i,j} \overline{e_j}(\lambda) = \boldsymbol{\alpha}^\top \mathbf{A}^x \boldsymbol{\Omega},
\end{aligned}$$

showing that the vv-WFA $(\boldsymbol{\alpha}, \{\mathbf{A}^\sigma\}_{\sigma\in\Sigma}, \boldsymbol{\Omega})$ computes $\vec{f}$ and consequently that $\operatorname{rank}(\vec{f}) \leq n = \operatorname{rank}(\boldsymbol{\mathcal{H}}_{(1)})$. □

A.2 Proof of Corollary 4

Corollary. *Let $\vec{f}: \Sigma^* \to \mathbb{R}^d$ be a recognizable function with rank n, let $\boldsymbol{\mathcal{H}} \in \mathbb{R}^{\Sigma^*\times d\times\Sigma^*}$ be its Hankel tensor, and for each $\sigma \in \Sigma$ let $\boldsymbol{\mathcal{H}}^\sigma \in \mathbb{R}^{\Sigma^*\times d\times\Sigma^*}$ be defined by $\boldsymbol{\mathcal{H}}^\sigma_{u,:,v} = f(u\sigma v)$ for all $u, v \in \Sigma^*$.*

Then, for any $\mathbf{P} \in \mathbb{R}^{\Sigma^*\times n}$ and $\boldsymbol{\mathcal{S}} \in \mathbb{R}^{n\times d\times\Sigma^*}$ such that $\boldsymbol{\mathcal{H}} = \boldsymbol{\mathcal{S}} \times_1 \mathbf{P}$, the vv-WFA $A = (\boldsymbol{\alpha}, \{\mathbf{A}^\sigma\}_{\sigma\in\Sigma}, \boldsymbol{\Omega})$ defined by $\boldsymbol{\alpha}^\top = \mathbf{P}_{\lambda,:}$, $\boldsymbol{\Omega} = \boldsymbol{\mathcal{S}}_{:,:,\lambda}$, and $\mathbf{A}^\sigma = \mathbf{P}^\dagger \boldsymbol{\mathcal{H}}^\sigma_{(1)} (\boldsymbol{\mathcal{S}}_{(1)})^\dagger$ is a minimal vv-WFA computing $\vec{f}$.

Proof. Let $\hat{A} = (\hat{\boldsymbol{\alpha}}^\top, \{\hat{\mathbf{A}}^\sigma\}_{\sigma\in\Sigma}, \hat{\boldsymbol{\Omega}})$ be a minimal vv-WFA computing $\vec{f}$ and let $\hat{\mathbf{P}} \in \mathbb{R}^{\Sigma^*\times n}$ and $\hat{\boldsymbol{\mathcal{S}}} \in \mathbb{R}^{n\times d\times\Sigma^*}$ be defined by

$$\hat{\mathbf{P}}_{u,:} = \boldsymbol{\alpha}^\top \hat{\mathbf{A}}^u \text{ and } \hat{\boldsymbol{\mathcal{S}}}_{:,:,v} = \hat{\mathbf{A}}^v \boldsymbol{\Omega}, \quad u, v \in \Sigma^*,$$

hence $\boldsymbol{\mathcal{H}} = \hat{\boldsymbol{\mathcal{S}}} \times_1 \hat{\mathbf{P}}$ and, equivalently, $\boldsymbol{\mathcal{H}}_{(1)} = \hat{\mathbf{P}}\hat{\boldsymbol{\mathcal{S}}}_{(1)}$. We will show that $\boldsymbol{\alpha}^\top = \hat{\boldsymbol{\alpha}}^\top \mathbf{M}^{-1}$, $\boldsymbol{\Omega} = \mathbf{M}\hat{\boldsymbol{\Omega}}$ and $\mathbf{A}^\sigma = \mathbf{M}\hat{\mathbf{A}}^\sigma\mathbf{M}^{-1}$ for each $\sigma \in \Sigma$ where $\mathbf{M} = \mathbf{P}^\dagger\hat{\mathbf{P}}$, which will imply $\vec{f}_A = \vec{f}_{\hat{A}} = \vec{f}$.

To simplify the notations, let $\mathbf{H} = \boldsymbol{\mathcal{H}}_{(1)}$, $\mathbf{S} = \boldsymbol{\mathcal{S}}_{(1)}$, $\hat{\mathbf{S}} = \hat{\boldsymbol{\mathcal{S}}}_{(1)}$, and $\mathbf{H}^\sigma = (\boldsymbol{\mathcal{H}}^\sigma)_{(1)}$ for each $\sigma \in \Sigma$. First observe that since $\mathbf{P}^\dagger\hat{\mathbf{P}}\hat{\mathbf{S}}\mathbf{S}^\dagger = \mathbf{P}^\dagger\mathbf{H}\mathbf{S}^\dagger = \mathbf{I}$, the matrix $\mathbf{M}$ is invertible with $\mathbf{M}^{-1} = \hat{\mathbf{S}}\mathbf{S}^\dagger$. Using the identities $\mathbf{H}^\sigma = \hat{\mathbf{P}}\hat{\mathbf{A}}^\sigma\hat{\mathbf{S}}$, $\mathbf{H}_{\lambda,:} = \hat{\boldsymbol{\alpha}}^\top\hat{\mathbf{S}}$, $\mathbf{P}^\dagger\boldsymbol{\mathcal{H}}_{:,:,\lambda} = \boldsymbol{\mathcal{S}}_{:,:,\lambda}$, and $\boldsymbol{\mathcal{H}}_{:,:,\lambda} = \hat{\mathbf{P}}\hat{\boldsymbol{\Omega}}$, we then get

$$\begin{aligned}
\mathbf{A}^\sigma &= \mathbf{P}^\dagger\mathbf{H}^\sigma\mathbf{S}^\dagger = \mathbf{P}^\dagger\hat{\mathbf{P}}\hat{\mathbf{A}}^\sigma\hat{\mathbf{S}}\mathbf{S}^\dagger = \mathbf{M}\hat{\mathbf{A}}^\sigma\mathbf{M}^{-1},\\
\boldsymbol{\alpha}^\top &= \mathbf{P}_{\lambda,:} = \mathbf{H}_{\lambda,:}\mathbf{S}^\dagger = \hat{\boldsymbol{\alpha}}^\top\hat{\mathbf{S}}\mathbf{S}^\dagger = \hat{\boldsymbol{\alpha}}^\top\mathbf{M}^{-1}, \text{ and}\\
\boldsymbol{\Omega} &= \boldsymbol{\mathcal{S}}_{:,:,\lambda} = \mathbf{P}^\dagger\boldsymbol{\mathcal{H}}_{:,:,\lambda} = \mathbf{P}^\dagger\hat{\mathbf{P}}\hat{\boldsymbol{\Omega}} = \mathbf{M}\hat{\boldsymbol{\Omega}}.
\end{aligned}$$

□

A.3 Proof of Proposition 1

Proposition. *Let $\vec{f} : \Sigma^* \to \mathbb{R}^d$ be a function computed by a vv-WFA $A = (\boldsymbol{\alpha}, \{\mathbf{A}^\sigma\}_{\sigma\in\Sigma}, \boldsymbol{\Omega})$ with n states and let $\mathcal{P}, \mathcal{S} \subseteq \Sigma^*$ be a complete basis for $\vec{f}$. For any $i \in [d]$, let $f_i : \Sigma^* \to \mathbb{R}$ be defined by $f_i(x) = \vec{f}(x)_i$ for all $x \in \Sigma^*$ and let n_i denote the rank of f_i.*

Let $\mathbf{P} \in \mathbb{R}^{\mathcal{P}\times n}$ be defined by $\mathbf{P}_{x,:} = \boldsymbol{\alpha}^\top\mathbf{A}^x$ for all $x \in \mathcal{P}$ and, for $i \in [d]$, let $\mathbf{H}_i = \mathbf{U}_i\mathbf{D}_i\mathbf{V}_i^\top$ be the thin SVD of $\mathbf{H}_i$ (i.e. $\mathbf{D}_i \in \mathbb{R}^{n_i\times n_i}$) where $\mathbf{H}_i \in \mathbb{R}^{\mathcal{P}\times\mathcal{S}}$ is the hankel matrix of f_i.

Then, for any $i \in [d]$, the WFA $A_i = \langle\boldsymbol{\alpha}_i, \{\mathbf{A}_i^\sigma\}_{\sigma\in\Sigma}\}, \boldsymbol{\omega}_i\rangle$ defined by

$$\boldsymbol{\alpha}_i^\top = \boldsymbol{\alpha}^\top\mathbf{P}^\dagger\mathbf{U}_i,\ \boldsymbol{\omega}_i = \mathbf{U}_i^\top\mathbf{P}\boldsymbol{\Omega}_{:,i} \text{ and } \mathbf{A}_i^\sigma = \mathbf{U}_i^\top\mathbf{P}\mathbf{A}^\sigma\mathbf{P}^\dagger\mathbf{U}_i \text{ for each } \sigma \in \Sigma,$$

is a minimal WFA computing f_i.

Proof. For each $i \in [d]$, let $\mathbf{S}_i \in \mathbb{R}^{n\times\mathcal{S}}$ be defined by $(\mathbf{S}_i)_{:,x} = \mathbf{A}^x\boldsymbol{\Omega}_{:,i}$ and consider the $|\mathcal{P}| \times d|\mathcal{S}|$ block matrices $\mathbf{H} = [\mathbf{H}_1, \ \cdots \ , \mathbf{H}_d]$, $\mathbf{H}^\sigma = [\mathbf{H}_1^\sigma, \ \cdots \ , \mathbf{H}_d^\sigma]$ for each $\sigma \in \Sigma$, and $\mathbf{S} = [\mathbf{S}_1, \ \cdots \ , \mathbf{S}_d]$.

We show the result for $i = 1$. First, it follows from applying Corollary 2 to the factorization $\mathbf{H}_1 = \mathbf{U}_1(\mathbf{D}_1\mathbf{V}_1^\top)$ that the WFA $\hat{A} = \langle\hat{\boldsymbol{\alpha}}, \{\hat{\mathbf{A}}^\sigma\}_{\sigma\in\Sigma}, \hat{\boldsymbol{\omega}}\rangle$ defined by

$$\hat{\boldsymbol{\alpha}}^\top = (\mathbf{U}_1)_{\lambda,:},\ \hat{\boldsymbol{\omega}} = (\mathbf{D}\mathbf{V}_1^\top)_{:,\lambda} \text{ and } \hat{\mathbf{A}}^\sigma = \mathbf{U}_1^\top\mathbf{H}_1^\sigma\mathbf{V}_1\mathbf{D}_1^{-1} \text{ for each } \sigma \in \Sigma$$

is a minimal WFA computing f_1. We will show that the WFA A_1 is exactly $\hat{A}$.

Let $\sigma \in \Sigma$. We start by showing that $\mathbf{A}_1^\sigma = \hat{\mathbf{A}}^\sigma$. It is easy to check that $\mathbf{H} = \mathbf{P}\mathbf{S}$ and $\mathbf{H}^\sigma = \mathbf{P}\mathbf{A}^\sigma\mathbf{S}$. Furthermore, since $\mathbf{H}^\sigma = \mathbf{H}^\sigma\mathbf{S}^\dagger\mathbf{S}$ we have $\mathbf{H}_1^\sigma = \mathbf{H}^\sigma\mathbf{S}^\dagger\mathbf{S}_1$, which implies $\mathbf{A}^\sigma\mathbf{S}_1 = \mathbf{P}^\dagger\mathbf{H}^\sigma\mathbf{S}^\dagger\mathbf{S}_1 = \mathbf{P}^\dagger\mathbf{H}_1^\sigma$. It then follows that

$$\begin{aligned}
\mathbf{A}_1^\sigma &= \mathbf{U}_1^\top\mathbf{P}\mathbf{A}^\sigma\mathbf{P}^\dagger\mathbf{U}_1\\
&= \mathbf{U}_1^\top\mathbf{P}\mathbf{A}^\sigma\mathbf{P}^\dagger\mathbf{U}_1(\mathbf{D}_1\mathbf{V}_1^\top\mathbf{V}_1\mathbf{D}_1^{-1})\\
&= \mathbf{U}_1^\top\mathbf{P}\mathbf{A}^\sigma\mathbf{P}^\dagger\mathbf{H}_1\mathbf{V}_1\mathbf{D}_1^{-1}\\
&= \mathbf{U}_1^\top\mathbf{P}\mathbf{A}^\sigma\mathbf{S}_1\mathbf{V}_1\mathbf{D}_1^{-1}\\
&= \mathbf{U}_1^\top\mathbf{P}\mathbf{P}^\dagger\mathbf{H}_1^\sigma\mathbf{V}_1\mathbf{D}_1^{-1}\\
&= \mathbf{U}_1^\top\mathbf{H}_1^\sigma\mathbf{V}_1\mathbf{D}_1^{-1} = \hat{\mathbf{A}}^\sigma
\end{aligned}$$

where we also used the fact that $\mathbf{P}\mathbf{P}^\dagger\mathbf{H}_1^\sigma = \mathbf{H}_1^\sigma$ and $\mathbf{P}^\dagger\mathbf{H}_1 = \mathbf{S}_1$. Now since the column space of $\mathbf{U}_1$ is contained in the column space of $\mathbf{P}$, we have $\mathbf{U}_1^\top\mathbf{P}\mathbf{P}^\dagger = \mathbf{U}_1^\top$ (and similarly $\mathbf{P}\mathbf{P}^\dagger\mathbf{U}_1 = \mathbf{U}_1$). Using the this fact and observing that $\boldsymbol{\alpha}^\top = \mathbf{H}_{\lambda,:}\mathbf{S}^\dagger$ and $\boldsymbol{\Omega} = \mathbf{P}^\dagger(\boldsymbol{\mathcal{H}}_{:,:,\lambda})$ we get

$$\boldsymbol{\alpha}_1^\top = \boldsymbol{\alpha}^\top\mathbf{P}^\dagger\mathbf{U}_1 = \mathbf{H}_{\lambda,:}\mathbf{S}^\dagger\mathbf{P}^\dagger\mathbf{U}_1 = \mathbf{P}_{\lambda,:}\mathbf{P}^\dagger\mathbf{U}_1 = (\mathbf{U}_1)_{\lambda,:} = \hat{\boldsymbol{\alpha}}^\top$$

and

$$\boldsymbol{\omega}_1 = \mathbf{U}_1^\top \mathbf{P}\boldsymbol{\Omega}_{:,1} = \mathbf{U}_1^\top \mathbf{P}\mathbf{P}^\dagger(\boldsymbol{\mathcal{H}}_{:,1,\lambda}) = \mathbf{U}_1^\top (\mathbf{H}_1)_{:,\lambda} = \hat{\boldsymbol{\omega}}$$

which concludes the proof. ☐

A.4 Proof of Theorem 5

Theorem. *Let $\mathbf{M} \in \mathbb{R}^{d_1 \times d_2}$ be of rank R and let $\hat{\mathbf{M}} = \mathbf{M} + \mathbf{E}$ where $\mathbf{E}$ is a random noise term such that $\frac{\mathrm{vec}(\mathbf{E})}{\|\mathbf{E}\|_F}$ follows a uniform distribution on the unit sphere in $\mathbb{R}^{d_1 d_2}$. Let $\boldsymbol{\Pi}_U, \boldsymbol{\Pi}_{\hat{U}} \in \mathbb{R}^{d_1 \times d_1}$ be the matrices of the orthogonal projections onto the space spanned by the top R left singular vectors of $\mathbf{M}$ and $\hat{\mathbf{M}}$ respectively.*

Let $\delta > 0$, let $\alpha = \mathfrak{s}_R(\mathbf{M})$ be the smallest non-zero singular value of $\mathbf{M}$ and suppose that $\|\mathbf{E}\|_F \leq \alpha/2$. Then, with probability at least $1 - \delta$,

$$\|\boldsymbol{\Pi}_U - \boldsymbol{\Pi}_{\hat{U}}\|_F \leq 4\left(\sqrt{\frac{(d_1 - R)R + 2\log(1/\delta)}{d_1 d_2}}\frac{\|\mathbf{E}\|_F}{\alpha} + \frac{\|\mathbf{E}\|_F^2}{\alpha^2}\right).$$

Let $\boldsymbol{\Pi}_{U_\perp} = \mathbf{I} - \boldsymbol{\Pi}_U$ and $\boldsymbol{\Pi}_{V_\perp} = \mathbf{I} - \boldsymbol{\Pi}_V$. Then, under the assumption $\|\mathbf{E}\|_F \leq \alpha/2$, it follows from Theorem 1 in [6] that

$$\|\boldsymbol{\Pi}_U - \boldsymbol{\Pi}_{\hat{U}}\|_F \leq \frac{2\sqrt{2}}{\alpha}\left(\|\boldsymbol{\Pi}_{U_\perp}\mathbf{E}\boldsymbol{\Pi}_V\|_F + \frac{\|\boldsymbol{\Pi}_{U_\perp}\mathbf{E}\boldsymbol{\Pi}_{V_\perp}\|_F \cdot \|\boldsymbol{\Pi}_U\mathbf{E}\boldsymbol{\Pi}_{V_\perp}\|_F}{\alpha}\right).$$

The second term of the sum can be bounded using the fact that both $\|\boldsymbol{\Pi}_{U_\perp}\mathbf{E}\boldsymbol{\Pi}_{V_\perp}\|_F$ and $\|\boldsymbol{\Pi}_U\mathbf{E}\boldsymbol{\Pi}_{V_\perp}\|_F$ are bounded by $\|\mathbf{E}\|_F$. Indeed we have e.g.

$$\|\boldsymbol{\Pi}_{U_\perp}\mathbf{E}\boldsymbol{\Pi}_{V_\perp}\|_F = \|(\boldsymbol{\Pi}_{V_\perp} \otimes \boldsymbol{\Pi}_{U_\perp})\mathrm{vec}(\mathbf{E})\|_F \leq \|\mathrm{vec}(\mathbf{E})\|_F = \|\mathbf{E}\|_F$$

since $\boldsymbol{\Pi}_{V_\perp} \otimes \boldsymbol{\Pi}_{U_\perp}$ is the matrix of an orthogonal projection. To bound the first term, we use the following lemma showing that the norm of a d-dimensional random vector $\mathbf{v}$ projected onto a fixed subspace of dimension k will be concentrated around $\sqrt{k/d}\|\mathbf{v}\|$.

Lemma 1. *Let $\boldsymbol{\Pi} \in \mathbb{R}^{d \times d}$ be a rank k projection matrix and let $\mathbf{v} \in \mathbb{R}^d$ be a random variable such that $\frac{\mathbf{v}}{\|\mathbf{v}\|}$ follows a uniform distribution on the unit sphere in $\mathbb{R}^d$. Then, for any $\delta > 0$,*

$$\mathbb{P}\left[\|\boldsymbol{\Pi}\mathbf{v}\|_2^2 > 2\frac{k + 2\log(1/\delta)}{d}\|\mathbf{v}\|_2^2\right] \leq \delta.$$

Proof. This directly comes form the following classical result (see e.g. Lemma 2.4 in [10]): if $\mathbf{x}$ is a random unit vector drawn uniformly from the unit sphere we have for any $\beta > 1$

$$\mathbb{P}\left[\|\boldsymbol{\Pi}\mathbf{x}\|_2^2 \leq \beta\frac{k}{d}\right] \leq \exp\left\{\frac{k}{2}(1 - \beta + \log\beta)\right\}.$$

Using the inequality $\log\beta \leq \beta/2$, the right term can be upper bounded by $\exp(k/2(1 - \beta/2)$, and by setting this upper bound equal to δ we get $\beta = 2(1 + 2\log(1/\delta)/k)$ which leads to the result. ☐

Applying this lemma to $\|\boldsymbol{\Pi}_{U_\perp}\mathbf{E}\boldsymbol{\Pi}_V\|_F = \|(\boldsymbol{\Pi}_V \otimes \boldsymbol{\Pi}_{U_\perp})\mathrm{vec}(\mathbf{E})\|_2$ by observing that $\boldsymbol{\Pi}_V \otimes \boldsymbol{\Pi}_{U_\perp}$ is a $d_1 d_2 \times d_1 d_2$ projection matrix of rank $R(d_1 - R)$, we get that $\|\boldsymbol{\Pi}_{U_\perp}\mathbf{E}\boldsymbol{\Pi}_V\|_F \leq \sqrt{2\frac{(d_1-R)R+2\log(1/\delta)}{d_1 d_2}}\|\mathbf{E}\|_F$ with probability at least $1 - \delta$ which concludes the proof.

B Detailed Results for Experiments on Real Data

The perplexity and WER on the test sets for all languages are reported in Table 2 when MT-SL is used with all other languages as related tasks, and in Table 3 when only the 4 closest languages are used. The list of the closest languages used for each task can be found in Table 4.

		Perplexity		Word Error Rate (%)	
Language	Training size	SL	MT-SL	SL	MT-SL
Ancient Greek	100	**4.038**	4.152	78.868	78.868
	500	4.119	4.140	77.735	78.108
	1000	4.239	4.207	74.661	74.923
	all	4.582	4.564	75.596	75.804
Arabic	100	2.298	2.291	78.320	**77.105**
	500	2.300	2.293	74.134	74.134
	1000	2.308	2.305	67.251	67.251
	all	2.306	2.338	66.595	66.595
Basque	100	6.184	6.196	75.398	**71.241**
	500	6.220	**6.067**	77.511	**71.157**
	1000	6.268	6.268	76.388	**68.452**
	all	6.760	6.760	75.803	**68.192**
Bulgarian	100	5.240	**5.121**	73.009	73.009
	500	5.475	5.475	67.561	67.733
	1000	5.616	5.616	66.315	**63.786**
	all	6.162	6.162	66.018	**62.196**
Croatian	100	5.621	**4.824**	74.566	74.358
	500	5.357	**4.998**	74.890	74.844
	1000	5.334	**5.148**	75.214	75.306
	all	5.285	5.260	77.850	**76.000**
Czech	100	4.248	**3.857**	80.417	**78.736**
	500	4.443	4.404	74.604	74.091
	1000	4.533	4.537	73.977	73.728
	all	5.091	5.091	73.849	**71.325**
Danish	100	5.028	**4.914**	79.890	**75.733**
	500	5.080	**4.932**	72.494	72.494
	1000	5.069	**4.939**	70.674	70.674
	all	5.363	**5.176**	70.674	70.674
Dutch	100	6.380	**5.925**	80.204	80.204
	500	**6.856**	7.209	74.158	74.158
	1000	**6.758**	7.223	73.924	73.924
	all	**8.025**	8.201	72.785	72.785
English	100	5.223	**5.065**	72.734	72.734
	500	5.748	**5.596**	72.197	72.197
	1000	5.764	5.808	70.371	69.400
	all	6.442	6.464	67.626	67.626
Estonian	100	**5.242**	5.835	50.874	50.874
	500	6.107	**5.682**	**48.666**	50.138
	1000	6.605	**6.289**	**49.862**	50.966
	all	6.653	**5.706**	50.046	50.414
Finnish	100	5.492	**4.655**	68.906	68.906
	500	5.974	**5.821**	68.442	67.557
	1000	6.146	**5.846**	66.237	66.237
	all	7.709	**7.420**	63.811	62.848
French	100	3.680	**3.291**	70.243	**65.828**
	500	3.674	**3.573**	62.685	62.685
	1000	3.724	3.677	62.220	61.755
	all	3.823	3.823	59.732	59.732
German	100	5.572	**4.961**	77.083	77.188
	500	5.676	**5.427**	76.637	76.637
	1000	5.740	5.740	74.514	74.514
	all	6.056	6.056	**72.554**	73.790
Gothic	100	6.120	6.120	80.046	**76.236**
	500	6.807	**6.590**	76.325	**73.720**
	1000	6.940	**6.562**	75.439	**73.755**
	all	7.777	**7.178**	74.074	**72.479**
Greek	100	4.186	**3.870**	66.813	66.813
	500	4.105	**3.917**	69.233	69.233
	1000	4.177	4.096	66.339	66.339
	all	4.088	3.997	67.203	66.695
Hebrew	100	3.953	**3.715**	71.615	71.615
	500	3.948	3.948	74.295	74.295
	1000	3.980	**3.856**	76.157	**72.757**
	all	4.022	3.945	73.359	73.359
Hindi	100	3.898	3.809	58.776	58.776
	500	4.219	**4.072**	62.645	**61.341**
	1000	4.095	4.095	61.381	61.381
	all	4.340	4.319	59.818	59.818
Hungarian	100	5.772	**5.048**	68.809	68.844
	500	5.766	**4.990**	69.787	69.857
	1000	5.579	**5.120**	69.193	69.053
	all	5.592	**5.147**	69.403	69.403
Indonesian	100	4.818	**4.302**	77.774	**74.451**
	500	4.650	**4.448**	70.560	70.560
	1000	4.639	**4.444**	70.682	70.682
	all	4.734	**4.614**	71.160	71.160
Irish	100	3.580	**3.434**	69.202	69.428
	500	3.543	3.491	66.885	66.885
	1000	3.594	3.559	66.885	66.885
	all	3.594	3.559	66.885	66.885
Italian	100	3.418	**3.235**	60.659	60.659
	500	3.408	**3.299**	57.976	57.976
	1000	3.480	**3.310**	57.906	57.748
	all	3.620	**3.506**	57.574	57.574
Japanese	100	3.087	**2.984**	63.968	63.702
	500	3.203	3.156	64.016	**61.722**
	1000	3.121	3.141	62.433	61.482
	all	3.196	3.221	61.837	**59.632**
Latin	100	4.800	4.784	82.052	81.377
	500	5.094	5.059	78.482	**76.479**
	1000	5.296	5.281	76.024	75.342
	all	6.241	6.239	75.179	**72.662**
Norwegian	100	5.070	**4.828**	75.248	**73.249**
	500	5.177	**4.927**	71.129	71.129
	1000	5.267	**5.163**	69.099	68.448
	all	5.733	**5.632**	69.487	**66.716**
Old Church Slavonic	100	6.017	6.003	73.649	**72.107**
	500	7.220	7.150	69.254	70.246
	1000	7.731	**7.552**	68.758	68.722
	all	8.889	**8.465**	68.067	**66.968**
Persian	100	3.218	**3.079**	66.111	**64.360**
	500	3.250	3.298	58.693	58.693
	1000	3.275	3.310	58.531	**57.093**
	all	3.339	3.339	58.164	**55.354**
Polish	100	4.618	**4.373**	68.314	68.314
	500	5.199	**5.086**	68.402	**66.380**
	1000	5.466	5.475	69.338	**64.724**
	all	6.404	**6.184**	63.802	63.802
Portuguese	100	4.119	**3.675**	72.949	72.949
	500	**3.977**	4.084	69.618	69.618
	1000	4.176	**4.052**	69.017	69.017
	all	4.288	4.342	68.757	**65.491**
Romanian	100	7.269	**5.405**	71.860	71.311
	500	7.269	**6.288**	70.075	69.664
	1000	7.269	**6.288**	70.075	69.664
	all	7.269	**6.288**	70.075	69.664
Slovenian	100	4.970	5.034	72.423	71.985
	500	5.199	5.163	71.238	**68.027**
	1000	5.591	**5.179**	70.242	**67.650**
	all	5.605	**5.406**	70.875	**64.943**
Spanish	100	3.138	3.068	67.485	67.485
	500	3.103	**2.984**	66.890	66.890
	1000	3.167	3.068	63.851	63.717
	all	3.265	3.176	64.702	64.702
Swedish	100	5.161	**4.946**	74.509	74.509
	500	5.278	**5.080**	73.143	**69.934**
	1000	5.511	**5.281**	71.004	**69.355**
	all	5.737	**5.489**	68.878	68.878
Tamil	100	8.651	**5.929**	66.999	66.667
	500	8.243	**6.149**	**64.296**	65.481
	1000	8.243	**6.149**	**64.296**	65.481
	all	8.243	**6.149**	**64.296**	65.481

Table 2: Detailed experimental results on the UNIDEP dataset when all other languages are used as related tasks.

Language	Training size	Perplexity		Word Error Rate (%)	
		SL	MT-SL	SL	MT-SL
Ancient Greek	100	4.084	4.064	81.332	**79.890**
	500	4.166	4.154	79.045	**77.993**
	1000	4.203	4.178	**77.802**	78.896
	all	4.582	4.612	**75.596**	76.850
Arabic	100	2.281	2.250	74.365	74.365
	500	2.306	2.278	69.322	69.322
	1000	2.318	2.291	68.583	68.583
	all	2.306	2.300	66.595	66.595
Basque	100	5.984	5.984	76.533	**71.780**
	500	6.120	**5.989**	76.701	**67.738**
	1000	6.231	6.170	76.052	**71.841**
	all	6.760	6.760	75.803	**69.064**
Bulgarian	100	5.115	**4.772**	69.294	69.294
	500	5.565	**5.327**	66.451	**64.902**
	1000	5.637	**5.432**	66.878	**64.231**
	all	6.162	6.162	66.018	**64.024**
Croatian	100	5.518	**4.695**	73.665	73.202
	500	5.429	**4.858**	75.723	**73.919**
	1000	5.278	**5.085**	78.127	**76.717**
	all	5.285	**5.163**	77.850	77.387
Czech	100	4.183	4.146	78.830	**76.088**
	500	4.471	4.487	75.772	75.114
	1000	4.588	4.563	73.770	73.770
	all	5.091	5.091	73.849	**71.693**
Danish	100	4.841	4.814	78.875	**77.264**
	500	4.906	4.870	76.410	**74.911**
	1000	5.114	5.192	70.964	71.012
	all	5.363	**5.203**	70.674	**69.562**
Dutch	100	6.875	**6.134**	74.644	74.644
	500	7.610	**7.310**	76.135	76.135
	1000	8.042	**7.483**	73.823	73.589
	all	8.025	8.062	72.785	72.098
English	100	5.269	**5.148**	73.919	73.919
	500	5.618	**5.502**	74.486	**70.485**
	1000	5.841	5.867	75.211	**69.153**
	all	**6.442**	6.653	67.626	67.538
Estonian	100	**5.404**	5.927	49.126	49.126
	500	6.112	**5.636**	50.046	50.046
	1000	6.607	6.613	50.138	50.414
	all	**6.653**	7.322	50.046	50.046
Finnish	100	5.706	**5.257**	69.102	**67.964**
	500	5.951	**5.538**	67.600	66.715
	1000	6.116	**5.777**	66.159	65.406
	all	7.709	**7.516**	63.811	63.558
French	100	3.878	**3.322**	69.163	68.494
	500	3.646	3.570	65.309	**62.397**
	1000	3.720	3.703	59.418	59.418
	all	3.823	3.800	**59.732**	60.785
German	100	5.843	**5.169**	78.226	**76.480**
	500	5.747	5.727	77.315	**72.769**
	1000	5.752	**5.565**	73.656	74.190
	all	6.056	6.056	72.554	72.264
Gothic	100	6.062	**5.774**	81.570	**75.226**
	500	6.701	**6.274**	77.583	**73.968**
	1000	6.955	**6.799**	74.659	**72.780**
	all	7.777	**7.187**	74.074	**72.391**
Greek	100	4.171	**3.654**	68.912	68.912
	500	4.094	**3.846**	68.810	68.810
	1000	4.188	**3.926**	66.543	66.543
	all	4.088	**3.964**	67.203	66.695
Hebrew	100	3.950	**3.777**	73.114	73.827
	500	3.959	**3.821**	77.814	77.814
	1000	3.975	**3.810**	74.770	74.770
	all	4.022	**3.918**	73.359	73.359
Hindi	100	4.118	4.026	62.440	**60.955**
	500	4.130	4.134	63.116	**60.082**
	1000	4.080	4.049	60.487	59.886
	all	4.340	4.344	59.818	59.341
Hungarian	100	5.724	**4.956**	69.647	70.171
	500	5.684	**5.039**	**68.949**	70.555
	1000	5.622	**5.107**	69.158	69.612
	all	5.592	**5.089**	69.403	69.752
Indonesian	100	4.716	**4.129**	75.472	74.475
	500	4.625	**4.267**	71.792	71.792
	1000	4.643	**4.463**	71.752	71.752
	all	4.734	**4.572**	71.160	71.160
Irish	100	3.679	**3.306**	66.457	66.457
	500	3.565	3.479	65.626	65.626
	1000	3.594	**3.425**	66.885	66.885
	all	3.594	**3.425**	66.885	66.885
Italian	100	3.366	**3.209**	66.725	**62.897**
	500	3.414	3.356	58.867	58.526
	1000	3.407	3.390	57.853	57.075
	all	3.620	3.575	57.574	57.276
Japanese	100	3.137	3.103	67.460	68.075
	500	3.077	3.099	63.224	63.224
	1000	3.143	3.179	61.185	60.887
	all	3.196	3.243	61.837	61.837
Latin	100	4.725	**4.616**	80.918	**77.025**
	500	5.128	5.097	77.432	**76.344**
	1000	5.261	5.227	76.406	**74.779**
	all	6.241	6.235	75.179	**73.198**
Norwegian	100	5.116	**4.940**	73.543	73.540
	500	5.239	**5.098**	71.276	70.997
	1000	5.277	5.230	69.152	69.152
	all	5.733	5.743	69.487	**68.132**
Old Church Slavonic	100	6.019	**5.803**	72.001	71.575
	500	7.157	**7.041**	69.024	68.988
	1000	7.749	**7.321**	69.644	**67.446**
	all	8.889	**8.662**	68.067	68.262
Persian	100	3.221	3.124	64.559	**62.459**
	500	3.240	3.325	57.989	57.989
	1000	3.375	3.358	58.597	**57.406**
	all	3.339	3.335	58.164	**56.100**
Polish	100	4.566	4.508	70.210	**63.208**
	500	5.168	**4.962**	67.644	**65.988**
	1000	5.437	**5.236**	68.820	**65.622**
	all	6.404	**6.048**	63.802	63.802
Portuguese	100	3.720	3.712	74.866	**68.903**
	500	3.966	3.986	71.308	**67.409**
	1000	4.049	**3.934**	68.822	**67.051**
	all	4.288	**4.113**	68.757	**65.053**
Romanian	100	7.105	**5.155**	69.115	69.389
	500	7.269	**5.936**	70.075	69.664
	1000	7.269	**5.936**	70.075	69.664
	all	7.269	**5.936**	70.075	69.664
Slovenian	100	5.231	**4.684**	76.267	**72.457**
	500	5.469	**4.957**	72.760	72.760
	1000	5.366	**4.953**	72.012	**69.117**
	all	5.605	**5.300**	70.875	**66.949**
Spanish	100	3.137	**3.024**	65.528	65.528
	500	3.139	**3.027**	66.221	**64.544**
	1000	3.160	**3.045**	64.276	64.276
	all	3.265	**3.137**	64.702	**62.514**
Swedish	100	5.098	**4.856**	74.347	74.347
	500	5.350	5.340	68.920	68.920
	1000	5.389	5.301	69.115	**67.758**
	all	5.737	**5.558**	68.878	**66.619**
Tamil	100	8.798	**6.011**	67.046	**65.908**
	500	8.243	**6.205**	**64.296**	65.576
	1000	8.243	**6.205**	**64.296**	65.576
	all	8.243	**6.205**	**64.296**	65.576

Table 3: Detailed experimental results on the UNIDEP dataset when the 4 closest languages are used as related tasks. The closest languages used for each target task are reported in Table 4.

Target task	4 closest tasks w.r.t. subspace distance (closest first)			
Ancient Greek	Old Church Slavonic	Latin	Gothic	Hungarian
Arabic	Czech	Polish	Persian	Slovenian
Basque	Finnish	Polish	Czech	Indonesian
Bulgarian	Czech	Norwegian	Finnish	Slovenian
Croatian	Estonian	Slovenian	Czech	Finnish
Czech	Finnish	Norwegian	Bulgarian	Danish
Danish	Norwegian	Swedish	English	Czech
Dutch	German	Norwegian	Danish	English
English	Norwegian	Danish	Italian	Swedish
Estonian	Finnish	Swedish	Norwegian	Polish
Finnish	Estonian	Czech	Swedish	Norwegian
French	Italian	Spanish	German	English
German	Dutch	Swedish	English	French
Gothic	Old Church Slavonic	Latin	Ancient Greek	Finnish
Greek	Swedish	Spanish	Czech	German
Hebrew	Portuguese	Norwegian	Czech	Danish
Hindi	Japanese	Croatian	Tamil	Persian
Hungarian	Danish	Ancient Greek	German	Portuguese
Indonesian	Finnish	Czech	Bulgarian	Norwegian
Irish	Polish	Czech	Greek	Arabic
Italian	English	French	Spanish	Dutch
Japanese	Hindi	Persian	Arabic	Tamil
Latin	Old Church Slavonic	Ancient Greek	Gothic	Finnish
Norwegian	Danish	English	Swedish	Czech
Old Church Slavonic	Latin	Gothic	Ancient Greek	Finnish
Persian	Japanese	Czech	Swedish	Finnish
Polish	Slovenian	Czech	Finnish	Estonian
Portuguese	Hebrew	Norwegian	Italian	Danish
Romanian	Finnish	Estonian	Norwegian	Czech
Slovenian	Polish	Czech	Danish	Swedish
Spanish	French	Italian	Portuguese	Greek
Swedish	Danish	Norwegian	Finnish	Estonian
Tamil	Finnish	Indonesian	Basque	Croatian

Table 4: Related tasks used in the UNIDEP experiment. The 4 closest tasks were selected using subspace distance (i.e. Frobenius norm of the difference between the orthogonal projection matrices) between the space spanned by the top 50 left singular vectors of their Hankel matrices. The common basis of prefixes/suffixes $(\mathcal{P}, \mathcal{S})$ for these Hankel matrices was obtained by taking the union of the 100 most frequent prefixes/suffixes for each task.