Computer Science and Information Systems 12(2):465–486 DOI: 10.2298/CSIS140713014M

k-Best Max-margin Approaches for Sequence Labeling

Dejan Mančev[1] and Branimir Todorović[2]

Faculty of Science and Mathematics
University of Niš, Višegradska 33, 18000 Niš, Serbia,
[1] dejan.mancev@pmf.edu.rs; [2] braminirtodorovic@yahoo.com

Abstract. Structured learning algorithms usually require inference during the training procedure. Due to their exponential size of output space, the parameter update is performed only on a relatively small collection built from the "best" structures. The k-best MIRA is an example of an online algorithm which seeks optimal parameters by making updates on k structures with the highest score at a time. Following the idea of using k-best structures during the learning process, in this paper we introduce four new k-best extensions of max-margin structured algorithms. We discuss their properties and connection, and evaluate all algorithms on two sequence labeling problems, the shallow parsing and named entity recognition. The experiments show how the proposed algorithms are affected by the changes of k in terms of the F-measure and computational time, and that the proposed algorithms can improve results in comparison to the single best case. Moreover, the restriction to the single best case produces a comparison of the existing algorithms.

Keywords: structured learning, sequence labeling, k-best approach, max-margin training

1. Introduction

Structured classification is a problem of learning a mapping from the input to the output structured objects, where the output structures incorporate different relationships among its classes. Its algorithms have been widely applied for solving sequence labeling problems where we need to assign a single label to each member of the observed sequence. Popular sequence labeling problems in natural language processing are part-of-speech tagging (assigning a linguistic category to each word), shallow parsing (detecting syntactic chunk labels) and named entity recognition (detecting entities in sentences).

The general idea of the k-best approach is to restrict the learning process from the structured space to its subset constructed only of its k elements with the highest scores, since considering all elements from the original space would lead to an intractable learning process in real time. While the influence of the parameter k to the learning process is not yet explored in theory, empirically it is shown that small values of k contribute to the learning process, with slight degradation of performance for larger values. For k equal to one, different learning methods are developed, while for higher values there is k-best Margin Infused Relaxed Algorithm (MIRA). Following the advantages of the k-best MIRA presented in [2], [8], [19], our motivation for the paper is to investigate different possibilities of incorporating k-best structures into the learning process with the aim to

get better results than the corresponding single best version, and to provide the theoretical analysis of the introduced algorithms. Thus, we introduce four new k-best versions to train the structural SVM as extensions of the popular single best algorithms: LaRank, SDM, passive-aggressive and perceptron algorithm.
The main contributions of the paper are:

- Introducing four new k-best extensions of max-margin structured classifiers with a discussion of their properties and connections.
- Providing theoretical analysis for the k-best versions. The restricted version of the k-best passive-aggressive algorithm is introduced in order to satisfy theoretical guarantees in terms of cumulative prediction loss similar to the single best case.
- Providing empirical evaluation of the introduced k-best versions on different datasets. Also, the restriction to the single best case provides a comparison of the existing algorithms.

The paper is organized as follows: we first discuss the related work, and then in Section 3, we define the basic notation and the primal-dual problem of max-margin structural classifiers. After reviewing k-best MIRA, in Section 4, we introduce new k-best extensions for structured classification. In Section 5, we present the results of k-best extensions on two sequence labeling tasks and conclude the paper in the last section.

2. Related work

Sequence labeling problems have been traditionally solved with hidden Markov models (HMMs) [9] by learning transition and emission probabilities and finding the optimal sequence by dynamic programming. The limitation of the reach feature representation in HMMs is overcome by its discriminative training such as Maximum entropy Markov models (MEMMs) [24] and conditional random fields [25], which also further avoid the label-bias problem in MEMMs. The combination of maximum margin discriminative training and the HMMs results in the Hidden Markov SVM [3], which is later generalized to the structured SVM [1]. All these discriminative training algorithms can be seen as a minimization of a differently chosen regularized loss function.

Since the size of the output space in structured data is usually exponential, a full optimization with exponentially many constraints is not possible in real time. There are several approaches to dealing with such optimization. Taskar et al. [5] present an equivalent polynomial-size formulation in the maximum margin Markov network (M^{3}N) by introducing marginal variables. Joachims et al. [4] use the cutting plane method on the equivalent formulation with one slack shared across all data and show that the dual problem has a sparse solution. Finding a small set of constraints that is sufficient to approximate a solution is usually done by increasing the working set of constraints through iterations. Balamurugan et al. [13] present the sequential dual method (SDM) for structural SVMs in which they sequentially add the constraint generated for the best structure and apply the sequential minimal optimization (SMO) [14] with additional heuristics to increase speed. Jaggi et al. [31] propose a block-coordinate version of Frank-Wolfe optimization as an online algorithm with an efficiently computed optimal step size which has a duality gap guarantee. Chang et al. [35] suggest a dual coordinate descent applied on L_2 loss which allows updating only one dual variable at a time. Bordes et al. [22] present the LaRank algorithm adapted to structured problems. After processing a new training example and adding the corresponding constraint, it goes back and reprocesses the old examples by optimizing them with the possibility of further increasing their working set of constraints, in order to get training through the data in one pass. There are also online algorithms adapted to the structured version, such as the perceptron [17] and the passive-aggressive algorithm [7], suitable for large-scale problems.

While all previously mentioned approaches make a single increment to their working set of constraints or just restrict the optimization to only one constraint, Crammer et al. [2] first proposed the k-best version of the Margin Infused Relaxed Algorithm (MIRA) as an online algorithm which restricts optimization not only to one but to k-best constraints. The algorithm is designed to traverse through training examples, find the k-best structures inside the example with the current parameters and then use these structures to update the parameters before moving to the next example. The idea is to adjust the parameters using new structures, but keep the changes as small as possible in order to hold previously obtained knowledge. At each example, after finding k outputs with the highest score, the algorithm minimizes the norm of parameter change while satisfying constraints on generated outputs. Even though the algorithm traverses online through examples, inside each example it performs full optimization subject to generated k constraints. The algorithm was tested on different problems. Crammer et al. [2] present the results on handwriting recognition, noun phrase chunking and named entity recognition (NER), showing that k-best MIRA performs as well as batch methods such as CRFs and M^{3}N, but converges more quickly after a few iterations. McDonald et al. [8] use k-best MIRA for dependency parsing, testing different values of k, and state that even small values of k are sufficient to achieve close to best performance. Gimpel and Cohen [19] also test k-best MIRA on NER and achieve similar results to the perceptron algorithm that were slightly better than the CRF, while MIRA gave better results on the part-of-speech (POS) tagging than perceptron for a reasonably small parameter k. Other related works include using a primal sub-gradient method for the averaged sum loss [20] optimization, which is closely related to the k-best variant of the Pegasos algorithm [34]. The k-best approach is also applied in confidence weighted methods [32] and used for confidence in committee organization [33], where k-best paths are used to extract the confidence of a particular label which is included in the learning process to improve the results.

3. Structural support vector machines

Let $\mathcal{X}$ be an input alphabet and $\mathscr{D} = ((\boldsymbol{x}^n, \boldsymbol{y}^n))_{n=1}^N$ a training set, where each $\boldsymbol{x}^n \in \mathcal{X}^{T_n}$ represents an input sequence of length T_n with the corresponding structure $\boldsymbol{y}^n$. The set of all possible structures over the sequence $\boldsymbol{x}^n$ is denoted by $\mathscr{Y}(\boldsymbol{x}^n)$ and $\mathscr{Y}_{-n} = \mathscr{Y}(\boldsymbol{x}^n) \setminus \boldsymbol{y}^n$. In further discussion, we focus on the sequence labeling problem where $\mathscr{Y}(\boldsymbol{x}^n) = \mathcal{Y}^{T_n}$ and $\mathcal{Y}$ represents a set of possible labels for an element of $\mathcal{X}$. However, everything presented in this paper, except for the discussion of the inference problem, will hold for a general case of $\mathscr{Y}(\boldsymbol{x}^n)$. The linear classification problem is to learn a function $h_{\mathbf{w}}$ from the data $\mathscr{D}$, which maps every sequence $\boldsymbol{x}$ over the alphabet $\mathcal{X}$ to an element of $\mathscr{Y}(\boldsymbol{x})$ in the following form

$$h_{\mathbf{w}}(\boldsymbol{x}) = \arg\max_{\boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x})} \mathbf{w}^{\mathsf{T}} \mathbf{F}(\boldsymbol{x}, \boldsymbol{y}), \tag{1}$$

where $\mathbf{F}(\boldsymbol{x}, \boldsymbol{y})$ represents a *global feature vector* measuring the compatibility of $\boldsymbol{x}$ and $\boldsymbol{y}$. One way of finding such a function is to estimate $\mathbf{w}$ via the max-margin approach, which minimizes L_2-regularized empirical risk on $\mathscr{D}$ [15] leading to the following formulation of the structural support vector machine in the primal space

$$\min_{\mathbf{w},\boldsymbol{\xi}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{n=1}^{N} \xi_n \tag{2}$$

$$\text{s.t. } \mathbf{w}^{\mathsf{T}} \mathit{\Delta}\mathbf{F}_n(\boldsymbol{y}) \geq L(\boldsymbol{y}^n, \boldsymbol{y}) - \xi_n, \quad \forall n, \forall \boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n) \tag{3}$$

where $\mathit{\Delta}\mathbf{F}_n(\boldsymbol{y}) = \mathbf{F}(\boldsymbol{x}^n, \boldsymbol{y}^n) - \mathbf{F}(\boldsymbol{x}^n, \boldsymbol{y})$. According to the constraints, the original sequence $\boldsymbol{y}^n$ should produce a greater score $\mathbf{w}^{\mathsf{T}}\mathbf{F}(\boldsymbol{x}^n, \boldsymbol{y}^n)$ then any other sequence at least for the size of the margin for that sequence. The cost function $L(\boldsymbol{y}^n, \boldsymbol{y})$, used for margin scaling, represents the cost of labeling the sequence $\boldsymbol{x}^n$ with $\boldsymbol{y}$ instead of $\boldsymbol{y}^n$. To handle a non-separable case, for each example $\boldsymbol{x}^n$ we assign non-negative slack variables ξ_n which control the penalty for its misclassification and the whole problem refers to N-slack formulation with margin scaling. Since $L(\boldsymbol{y}^n, \boldsymbol{y}^n)$ is equal to zero, the constraint for the sequence $\boldsymbol{y}^n$ in (3) produces $\xi_n \geq 0$. The parameter $C \in \mathbb{R}^+$ controls the trade-off between the slack variable penalty and the size of the margin. As the cardinality of $\mathscr{Y}(\boldsymbol{x}^n)$ is exponential, the direct optimization is untraceable. A straightforward transformation of the primal problem (2)-(3), by introducing Lagrange multipliers $\boldsymbol{\lambda} = [\lambda_{n,\boldsymbol{y}}]_{n,\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)}$, leads to the dual optimization problem

$$\min_{\boldsymbol{\lambda}} \frac{1}{2}\boldsymbol{\lambda}^{\mathsf{T}}\boldsymbol{K}\boldsymbol{\lambda} - \boldsymbol{\lambda}^{\mathsf{T}}\mathbf{L} \tag{4}$$

$$\text{s.t.} \sum_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} \lambda_{n,\boldsymbol{y}} = C, \; \forall n, \quad \lambda_{n,\boldsymbol{y}} \geq 0, \; \forall n, \; \forall \boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n) \tag{5}$$

where $\boldsymbol{K}$ is a kernel matrix defined with elements $K_{n,\boldsymbol{y},m,\boldsymbol{y}'} = \mathit{\Delta}\mathbf{F}_n(\boldsymbol{y})^{\mathsf{T}}\mathit{\Delta}\mathbf{F}_m(\boldsymbol{y}')$ for every $\boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n)$, $\boldsymbol{y}' \in \mathscr{Y}(\boldsymbol{x}^m)$ and $\mathbf{L}$ is a vector with elements $L_{n,\boldsymbol{y}} = L(\boldsymbol{y}^n, \boldsymbol{y})$. The primal parameters are expressed in terms of dual ones as

$$\mathbf{w} = \sum_{n=1}^{N} \sum_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} \lambda_{n,\boldsymbol{y}} \mathit{\Delta}\mathbf{F}_n(\boldsymbol{y}). \tag{6}$$

The constraints in (5) allow us to perform the optimization restricted to a single example at a time [23]. Let $\boldsymbol{\alpha}_n$ denote changes in $\boldsymbol{\lambda}_n$ during the processing of the nth example, i.e. let the change of parameters be made according to the update $\lambda'_{n,\boldsymbol{y}} \leftarrow \lambda_{n,\boldsymbol{y}} + \alpha_{n,\boldsymbol{y}}$. In order for new dual parameters $\boldsymbol{\lambda}'_n$ to be feasible, the sum of $\alpha_{n,\boldsymbol{y}}$ should be zero as well as $\lambda'_{n,\boldsymbol{y}} \geq 0$. By dropping all terms in (4)-(5) which do not depend on $\boldsymbol{\alpha}_n$, we can rewrite this optimization restricted to the nth example with respect to the parameter change

$$\min_{\boldsymbol{\alpha}_n} \frac{1}{2}\boldsymbol{\alpha}_n^{\mathsf{T}}\boldsymbol{K}_n\boldsymbol{\alpha}_n - \boldsymbol{\alpha}_n^{\mathsf{T}}\boldsymbol{\ell}_n \tag{7}$$

$$\text{s.t.} \sum_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} \alpha_{n,\boldsymbol{y}} = 0, \quad \lambda_{n,\boldsymbol{y}} + \alpha_{n,\boldsymbol{y}} \geq 0, \; \forall \boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n) \tag{8}$$

where the vector $\boldsymbol{\ell}_n = [\ell(\mathbf{w}; (\boldsymbol{x}^n, \boldsymbol{y}))]_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)}$ is defined with $\ell(\mathbf{w}; (\boldsymbol{x}^n, \boldsymbol{y})) = L(\boldsymbol{y}^n, \boldsymbol{y}) - \mathbf{w}^{\mathsf{T}} \Delta\mathbf{F}_n(\boldsymbol{y})$ and $\boldsymbol{K}_n$ is the block of the kernel matrix $\boldsymbol{K}$ corresponding to the nth example. Note that $\boldsymbol{\ell}_n$ depends on $\mathbf{w}$, which corresponds to parameters $\lambda_{n,\boldsymbol{y}}$ according to (6), and thus the parameters after the update will be

$$\mathbf{w}' = \mathbf{w} + \sum_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} \alpha_{n,\boldsymbol{y}} \Delta\mathbf{F}_n(\boldsymbol{y}). \tag{9}$$

The gradient of the dual function (7) is $\boldsymbol{g}(\boldsymbol{\alpha}) = [g_{n,\boldsymbol{y}}(\boldsymbol{\alpha})]$, with elements

$$g_{n,\boldsymbol{y}}(\boldsymbol{\alpha}) = \sum_{\boldsymbol{z}\in\mathscr{Y}(\boldsymbol{x}^n)} \alpha_{n,\boldsymbol{z}} K_{n,\boldsymbol{y},\boldsymbol{z}} - \ell(\mathbf{w}; (\boldsymbol{x}^n, \boldsymbol{y})), \tag{10}$$

which is used to express the violation of Karush-Kuhn-Tucker (KKT) conditions [11] for the problem (7)-(8) as

$$\max_{\boldsymbol{y}\in I_0} g_{n,\boldsymbol{y}}(\boldsymbol{\alpha}) > \min_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} g_{n,\boldsymbol{y}}(\boldsymbol{\alpha}) \tag{11}$$

where $I_0 = \{\boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n) : \alpha_{n,\boldsymbol{y}} > -\lambda_{n,\boldsymbol{y}}\}$. Again, the dual problem has exponentially many constraints.

4. k-best extensions

In this section we will describe extensions to the k-best case of popular algorithms. We start with reviewing the k-best MIRA from [2] and then introduce the k-best version of the perceptron, LaRank algorithm, sequential dual method, and passive-aggressive algorithm. We also introduce the restricted version of the passive-aggressive algorithm for which the cumulative prediction loss is bound in a similar way as in single best version.

4.1. k-best MIRA

Crammer et al. [16] propose the Margin Infused Relaxed Algorithm (MIRA) originally for multiclass problems, which was later extended to structured classification [2]. MIRA is defined in an online manner. After receiving a new example $(\boldsymbol{x}^n, \boldsymbol{y}^n)$, starting from parameters $\mathbf{w}_n$, the algorithm is searching for new parameters $\mathbf{w}$ in order to keep the norm of changing parameters as small as possible during the satisfaction of particular constraints on the received example. In a special case, the constraints on the example can be restricted to only one constraint corresponding to the predicted structure $\widetilde{\boldsymbol{y}}$ with the involved cost function. This is known as 1-best variant of MIRA, which is also called the online passive-aggressive (PA) algorithm [7]. The best sequence for $\boldsymbol{x}^n$ is found with respect to the loss function as

$$\widetilde{\boldsymbol{y}} = \arg\max_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} \ell(\mathbf{w}; (\boldsymbol{x}^n, \boldsymbol{y})) = \arg\max_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} \mathbf{w}^{\mathsf{T}} \mathbf{F}(\boldsymbol{x}^n, \boldsymbol{y}) + L(\boldsymbol{y}^n, \boldsymbol{y}). \tag{12}$$

MIRA was primarily designed for multiclass classification. In that case it belongs to the family of ultraconservative online algorithms [16], which consider only the updates of parameters corresponding to violated examples, i.e. to those examples that produce a

Algorithm 1: k-best MIRA

```
Input  : Training data: $\mathscr{D} = ((\boldsymbol{x}^n, \boldsymbol{y}^n))_{n=1}^N$, parameter $C \in \mathbb{R}^+$, $k \in \mathbb{N}$
         Number of iterations: $P$
Output: Model parameters: $\mathbf{w}$
$\mathbf{w} \leftarrow \mathbf{0}$;
for $p \leftarrow 1$ to $P$ do
    foreach $(\boldsymbol{x}^n, \boldsymbol{y}^n)$ in $\mathscr{D}$ do
        $S_n \leftarrow \mathscr{B}^k_{\mathbf{w}}$;                    /* find $k$-best sequences */
        Find $\mathbf{v}$ as a solution of the optimization problem:
            $\min_{\mathbf{v},\boldsymbol{\xi}} \frac{1}{2}\|\mathbf{v} - \mathbf{w}\|^2 + C \sum_{\boldsymbol{y} \in S_n} \xi_{n,\boldsymbol{y}}$
            s.t. $\mathbf{v}^\mathsf{T} \varDelta \mathbf{F}_n(y) \geq L(\boldsymbol{y}^n, \boldsymbol{y}) - \xi_{n,\boldsymbol{y}},\ \xi_{n,\boldsymbol{y}} \geq 0, \quad \forall \boldsymbol{y} \in S_n$
        $\mathbf{w} \leftarrow \mathbf{v}$;
```

higher score than the original one. Working with the ultraconservative property in the case of online structured classification leads to problems in optimization, due to the possibility of a large number of violated constraints. This produces the need to consider the optimization restricted only to the constraints for the first k sequences with the highest score, leading to the following optimization problem[1]

$$\min_{\mathbf{w},\xi_n} \frac{1}{2}\|\mathbf{w} - \mathbf{w}_n\|^2 + C \sum_{\boldsymbol{y} \in \mathscr{B}^k_{\mathbf{w}_n}} \xi_{n,\boldsymbol{y}} \tag{13}$$

$$\text{s.t. } \mathbf{w}^\mathsf{T} \varDelta \mathbf{F}_n(\boldsymbol{y}) \geq L(\boldsymbol{y}^n, \boldsymbol{y}) - \xi_{n,\boldsymbol{y}},\ \xi_{n,\boldsymbol{y}} \geq 0,\ \forall \boldsymbol{y} \in \mathscr{B}^k_{\mathbf{w}_n} \tag{14}$$

where $\mathscr{B}^k_{\mathbf{w}_n}$ represents the *k-best set*, formally defined as the set $\mathscr{B}^k_{\mathbf{w}_n} \subset \mathscr{Y}(\boldsymbol{x}^n)$ that satisfies $\forall \boldsymbol{y}' \in \mathscr{B}^k_{\mathbf{w}_n}, \forall \boldsymbol{y}'' \notin \mathscr{B}^k_{\mathbf{w}_n} \implies \ell(\mathbf{w}_n; (\boldsymbol{x}^n, \boldsymbol{y}')) \geq \ell(\mathbf{w}_n; (\boldsymbol{x}^n, \boldsymbol{y}''))$ and $|\mathscr{B}^k_{\mathbf{w}_n}| = \min\{k, |\mathscr{Y}(\boldsymbol{x}^n)|\}$. Note that with this formulation the set is not uniquely defined. The k-best MIRA is a general method which incorporates the k-best search in an online manner through examples without any specification for solving the optimization problem (13)-(14) inside the example. The pseudocode is presented in Algorithm 1.

4.2. *k*-best SDM for structural *SVMs*

In order to deal with a large dimensional problem (4)-(5), the sequential dual method (*SDM*) for structural *SVMs* [13] makes an assumption that at optimum only a small number of constraints inside one example is active. It considers the scaled version of the following sub-problem

$$\min_{\boldsymbol{\alpha}_n} \frac{1}{2} \boldsymbol{\alpha}_n^\mathsf{T} \mathbf{K}_n \boldsymbol{\alpha}_n - \boldsymbol{\alpha}_n^\mathsf{T} \boldsymbol{\ell}_n \tag{15}$$

$$\text{s.t. } \sum_{\boldsymbol{y} \in S_n} \alpha_{n,\boldsymbol{y}} = 0, \quad \alpha_{n,\boldsymbol{y}} \geq -\lambda_{n,\boldsymbol{y}}, \quad \forall \boldsymbol{y} \in S_n \tag{16}$$

[1] Crammer et al. in [2] define k-best MIRA for a linear separable problem and it is a straightforward extension to the linear non-separable case done by introducing slack variables $\xi_{n,\boldsymbol{y}}$, which we used here and also referred to it as k-best MIRA.

where the set of examples $S_n = \{\boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n) \ : \ \lambda_{n,\boldsymbol{y}} > 0\}$ is incrementally built by adding the sequence $\widetilde{\boldsymbol{y}}$ according to (12) that violates the margin the most on the current nth example and $\boldsymbol{K}_n$ denotes a block of matrix $\boldsymbol{K}$ which corresponds to the nth example. Specific heuristics are employed to control the growth of the set S_n in order to keep the number of optimization constraints at a reasonable size. After reaching the desired precision inside one example, the procedure continues to the next example to optimize, where the optimization is performed by the sequential minimal optimization (SMO) [14].

Now we will describe the SDM extension to the k-best case. The algorithm traverses through training examples and, at the nth example, its working set S_n is extended with current k-best sequences including the original sequence $\boldsymbol{y}^n$. Then the SMO is applied to optimize dual variables associated with sequences in S_n. It selects a pair of sequences from S_n based on the 'maximum violating pair' strategy and performs the optimization until the KKT conditions become satisfied on S_n. The test of the KKT condition violation, up to the precision τ, is

$$g_{n,\boldsymbol{y}''}(\boldsymbol{\alpha}) > g_{n,\boldsymbol{y}'}(\boldsymbol{\alpha}) + \tau, \tag{17}$$

$$\boldsymbol{y}' = \arg\min_{\boldsymbol{y}\in S_n} g_{n,\boldsymbol{y}}(\boldsymbol{\alpha}), \quad \boldsymbol{y}'' = \arg\max_{\boldsymbol{y}\in I_0'} g_{n,\boldsymbol{y}}(\boldsymbol{\alpha}), \tag{18}$$

where $I_0' = \{\boldsymbol{y} \in S_n : \alpha_{n,\boldsymbol{y}} > -\lambda_{n,\boldsymbol{y}}\}$. The gradient of the dual function is expressed by (10), where the sum over all elements of $\mathscr{Y}(\boldsymbol{x}^n)$ can be reduced for the sub-problem (15)-(16) to the sum over the elements from S_n as

$$g_{n,\boldsymbol{y}}(\boldsymbol{\alpha}) = \sum_{\boldsymbol{z}\in S_n} \alpha_{n,\boldsymbol{z}} K_{n,\boldsymbol{y},\boldsymbol{z}} - \ell(\mathbf{w}; (\boldsymbol{x}^n, \boldsymbol{y})) \tag{19}$$

since $\alpha_{n,\boldsymbol{y}}$ is equal to zero if $\boldsymbol{y}$ does not belong to S_n. After selecting sequences $\boldsymbol{y}'$ and $\boldsymbol{y}''$ which violate KKT conditions the most using (18), we consider the updates of the corresponding alpha parameters as $\alpha_{n,\boldsymbol{y}'} \leftarrow \alpha_{n,\boldsymbol{y}'} + \delta$ and $\alpha_{n,\boldsymbol{y}''} \leftarrow \alpha_{n,\boldsymbol{y}''} - \delta$. The optimization, now restricted to a single variable δ which must change the parameters so that they remain feasible, can be expressed in the analytical form as

$$\delta = \max\left(-\lambda_{n,\boldsymbol{y}'} - \alpha_{n,\boldsymbol{y}'}, \min\left(\lambda_{n,\boldsymbol{y}''} + \alpha_{n,\boldsymbol{y}''}, \frac{g_{n,\boldsymbol{y}''}(\boldsymbol{\alpha}) - g_{n,\boldsymbol{y}'}(\boldsymbol{\alpha})}{\|\Delta\mathbf{F}_n(\boldsymbol{y}') - \Delta\mathbf{F}_n(\boldsymbol{y}'')\|^2}\right)\right). \tag{20}$$

Note that after satisfying the KKT conditions for the set S_n and leaving that example, unsatisfied conditions inside the example could still be possible, since y' is found by minimization over S_n instead of $\mathscr{Y}(\boldsymbol{x}^n)$. As a time consuming operation, the minimization over whole $\mathscr{Y}(\boldsymbol{x}^n)$ is done only when we extend the current set S_n by k new sequences. These are the sequences corresponding to the lowest elements of the gradient of the dual function. Since S_n is extended when we start processing the nth example, all $\alpha_{n,\boldsymbol{y}}$ will be zero and thus these sequences are those with the highest $\ell(\mathbf{w}; (\boldsymbol{x}^n, y))$, e.i. the sequences from $\mathscr{B}^k_{\mathbf{w}}$.

In the end of the section, let us consider the difference between k-best MIRA optimized in dual space with the SMO and k-best SDM. As a consequence of a different primal formulation, when it returns to an earlier example k-best MIRA will create a new working set S_n while k-best SDM will update its previous set with new sequences (see Algorithm 2). Thus only the first epoch will be the same. However, if we consider the

1-best version of these algorithms, MIRA will be equivalent to the PA algorithm, while the SDM will have the first epoch through the data equivalent to MIRA and thus to the PA algorithm. These equivalents can also be observed because 1-best SDM in the first pass will choose sequences $\boldsymbol{y}' = \widetilde{\boldsymbol{y}}$ and $\boldsymbol{y}'' = \boldsymbol{y}^n$ on the nth example to optimize, and as all $\alpha_{n,\boldsymbol{y}}$ are zero at that moment and all $\lambda_{n,\boldsymbol{y}}$ are zero, except for $\lambda_{n,\boldsymbol{y}^n} = C$, the step δ in (20) will be reduced to the step from (21). However, note that this equivalence considers setting τ to zero, but in practical application τ is set to be a small positive tolerance because of numerical errors and the runtime of the algorithm.

4.3. k-best LaRank algorithm

The LaRank algorithm [21], introduced for multi-class problems, is a batch algorithm which uses the SMO with specific operations describing how to select a pair of coefficients for optimization. The algorithm has also been used for a structured output with a simpler scheduling for selecting a pair for optimization in an online manner and this version is called OLaRank [22]. We adopt the following notation: if an example $[(\boldsymbol{x}^n, \boldsymbol{y})]_{\boldsymbol{y} \in \mathscr{Y}(x^n)}$ has all dual variables $\lambda_{n,\boldsymbol{y}} = 0$ equal to zero we will call it *new example*, otherwise we will refer to it as an *old example* or *support pattern*. The algorithm with the exact inference is used to solve the dual problem (4)-(5) by applying three basic operations:

PROCESSNEW Pick a new example $[(\boldsymbol{x}^n, \boldsymbol{y})]$ and choose the pair of coefficients $\lambda_{n,\boldsymbol{y}^n}$ and $\lambda_{n,\widetilde{\boldsymbol{y}}}$ to optimize, where $\widetilde{\boldsymbol{y}}$ is found by (12),

PROCESSOLD Random pick a support pattern $[(\boldsymbol{x}^n, \boldsymbol{y})]$ and choose the best pair of coefficients $\lambda_{n,\boldsymbol{z}}$ and $\lambda_{n,\widetilde{\boldsymbol{y}}}$ to optimize according to the gradient vector,

OPTIMIZE Random pick a support pattern and choose the best pair of non-zero coefficients to optimize according to the gradient vector.

OLaRank applies the PROCESSNEW step followed by a specific number of n_R REPROCESS operations, where REPROCESS means applying ten OPTIMIZE steps after one PROCESSOLD step. Note that $n_R = 0$ reduces OLaRank to the previously described SDM for the first epoch.

We introduced the k-best version of the previous algorithm, calling it k-best LaRank, which optimizes the same dual problem with the SMO, which in a special case reduces to the k-best SDM (see Fig. 1). k-best LaRank uses the following k-best variants (kBVs) of operations:

kBV-PROCESSNEW Pick a new example $[(\boldsymbol{x}^n, \boldsymbol{y})]$ and optimize coefficients which correspond to its k-best sequences together with the original sequence $\boldsymbol{y}^n$.

kBV-PROCESSOLD Random pick a support pattern and optimize non-zero coefficients together with new coefficients from k-best sequences.

kBV-OPTIMIZE Random pick a support pattern and perform the optimization inside this pattern among non-zero coefficients,

with the same scheduling as in OLaRank. Let the optimization be performed inside a pattern while the KKT conditions are violated more than τ, as in (17)-(18), by choosing the sequence $\boldsymbol{y}''$ among the $\boldsymbol{y}$s for which $\alpha_{n,\boldsymbol{y}} > -\lambda_{n,\boldsymbol{y}} + \kappa$, where τ and κ are small positive tolerances. Similar to the single best case, we can bound the number of support vectors that k-best LaRank will add during the training procedure.

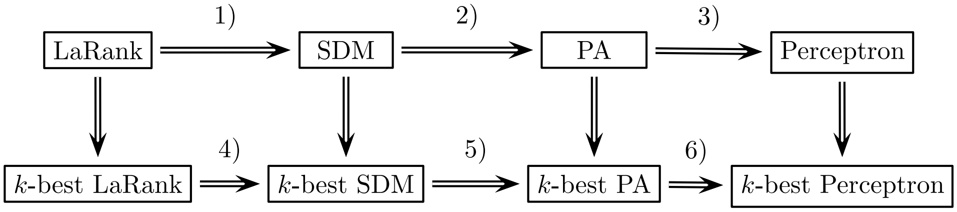

Fig. 1. Connections between learning algorithms and their k-best extensions for the first epoch.
1) and 4) $n_R = 0$. 2) Described in the end of Section 4.2.
3) and 6) No cost ($\mathbf{L} = \mathbf{0}$), fixed step size ($\delta = 1$). The connection holds for more than one epoch.
5) Performs one SMO step on each k-best sequence inside an example in sequential order.

Proposition 1. *During the first pass through the data with N examples, the k-best LaRank will add no more than*

$$\min\left\{N((n_R+1)k+1), \max\{\frac{2\rho NC}{\tau^2}, \frac{2NC}{\kappa\tau}\}\right\}$$

support vectors, where $\rho = \max_{n,\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} \|\Delta\mathbf{F}_n(\boldsymbol{y})\|^2$.

Proof. The first term follows directly from scheduling because each of kBV-ProcessNew operations can add up to $k+1$ support vectors, kBV-ProcessOld up to k support vectors, n_R times after each new example, while kBV-Optimize cannot add new support vectors. No matter what k we choose, in three basic k-best LaRank operations all SMO steps are performed only along the $\kappa\tau$-violation direction. Thus the k-best version also belongs to the Approximate Stochastic Witness Direction Search (see appendix in [26]), which reaches the $\kappa\tau$-optimality solution with no more than $\max\{\frac{2\rho NC}{\tau^2}, \frac{2NC}{\kappa\tau}\}$ SMO operations [22] [21], i.e. support vectors.

Bordes et al. [22] express the regret bound of LaRank ProcessNew operation through the data with the same value as the bound for the passive-aggressive algorithm. As neither ProcessOld nor Optimize operation can decrease the dual function, the same bound is stated for the whole LaRank algorithm. Since the previous two operations are not considered, the true regret should be much smaller. Similar results can be stated for the k-best version. As the first SMO step in kBV-ProcessNew will increase the dual function by the same value as ProcessNew, and as other SMO steps cannot decrease the dual function, as well as kBV-ProcessOld and kBV-Optimize operations, the k-best LaRank holds the same regret bound as the passive-aggressive algorithm. The pseudo code is presented in Algorithm 3.

Algorithm 2: k-best SDM

```
Input  : Training data: $D = ((\boldsymbol{x}^n, \boldsymbol{y}^n))_{n=1}^N$, parameter: $C \in \mathbb{R}^+$, $k \in \mathbb{N}$
         Number of iterations: $P$
Output: Model parameters: $\mathbf{w}$
$\mathbf{w} \leftarrow \mathbf{0}$; $\boldsymbol{\lambda} \leftarrow \mathbf{0}$;
$S_n \leftarrow \{\boldsymbol{y}^n\}$, $\lambda_{n,\boldsymbol{y}^n} \leftarrow C, \forall n = 1, \dots, N$;
for $p \leftarrow 1$ to $P$ do
    for $n \leftarrow 1$ to $N$ do
        $k$BV-ProcessNew($n$, $S_n$, $\mathbf{w}$, $\boldsymbol{\lambda}$);
```

Algorithm 3: k-best LaRank

```
Input  : Training data: $D = ((\boldsymbol{x}^n, \boldsymbol{y}^n))_{n=1}^N$, parameter: $C \in \mathbb{R}^+$, $k \in \mathbb{N}$
Output: Model parameters: $\mathbf{w}$
$\mathbf{w} \leftarrow \mathbf{0}$; $\boldsymbol{\lambda} \leftarrow \mathbf{0}$;
$S_n \leftarrow \{\boldsymbol{y}^n\}$, $\lambda_{n,\boldsymbol{y}^n} \leftarrow C, \forall n = 1, \dots, N$;
for $n \leftarrow 1$ to $N$ do
    $k$BV-ProcessNew($n$, $S_n$, $\mathbf{w}$, $\boldsymbol{\lambda}$);
    for $k \leftarrow 1$ to $n_R$ do
        $k$BV-Reprocess($(S_1, \dots, S_n)$, $\mathbf{w}$, $\boldsymbol{\lambda}$);
```

Procedure kBV-ProcessNew(n, S_n, $\mathbf{w}$, $\boldsymbol{\lambda}$)

```
$S_n \leftarrow S_n \cup \mathscr{B}_{\mathbf{w}}^k$; /* find $k$-best sequences of the $n$th example */
SMO($n$, $S_n$, $\mathbf{w}$, $\boldsymbol{\lambda}$);
```

Procedure kBV-Reprocess($(S_1, \dots, S_n)$, $\mathbf{w}$, $\boldsymbol{\lambda}$)

```
$k$BV-ProcessOld($(S_1, \dots, S_n)$, $\mathbf{w}$, $\boldsymbol{\lambda}$);
for $i \leftarrow 1$ to $maxIter$ do
    $k$BV-Optimize($(S_1, \dots, S_n)$, $\mathbf{w}$, $\boldsymbol{\lambda}$);
```

Procedure kBV-ProcessOld($(S_1, \dots, S_n)$, $\mathbf{w}$, $\boldsymbol{\lambda}$)

```
$m \leftarrow$ Random$[1, n]$;
$S_m \leftarrow S_m \cup \mathscr{B}_{\mathbf{w}}^k$;          /* find $k$-best sequences of the $m$th example */
SMO($m$, $S_m$, $\mathbf{w}$, $\boldsymbol{\lambda}$);
```

Procedure kBV-Optimize($(S_1, \dots, S_n)$, $\mathbf{w}$, $\boldsymbol{\lambda}$)

```
$m \leftarrow$ Random$[1, n]$;
SMO($m$, $S_m$, $\mathbf{w}$, $\boldsymbol{\lambda}$);
```

4.4. Passive-aggressive algorithms for k-best structural learning

The passive-aggressive (PA) is an online algorithm introduced in [7] and it solves the optimization problem (13)-(14) for $k = 1$, i.e. by using only the constraint generated from the best structure. We will refer to this algorithm as 1-best passive-aggressive or 1-best MIRA. Let us define the set of misclassified sequences with given parameters $Err_{\mathbf{w}_n} = \{\boldsymbol{y} \in \mathscr{Y}_{-n} : \ell(\mathbf{w}_n; (\boldsymbol{x}^n, \boldsymbol{y})) > 0\}$. This set will be used to decide weather or not the sequence should be used for parameter change. In order to calculate the step size, Crammer et al. [7] used the method of Lagrange multipliers on the problem (13)-(14), which for the single best version has only two constraints: the constraint for the structure with the highest score $\widetilde{\boldsymbol{y}}$ and the one for the original structure $\boldsymbol{y}^n$. The optimal step size is then found in the closed form as

$$\delta = \min \left\{ \frac{\max\{0, \ell(\mathbf{w}_n; (\boldsymbol{x}^n, \widetilde{\boldsymbol{y}}))\}}{\|\varDelta\mathbf{F}_n(\widetilde{\boldsymbol{y}})\|^2}, C \right\} . \tag{21}$$

The 1-best passive-aggressive algorithm considers updating the parameters only for the best sequence with the following behaviors as described in [7]:
i) *passive behavior* if the constraint in (14) for the best sequence of the current example is satisfied, the algorithm is passive, making no update;
ii)*aggressive behavior* if it is not a case, the algorithm makes an aggressive update

$$\mathbf{w} = \mathbf{w}_n + \delta\varDelta\mathbf{F}_n(\widetilde{\boldsymbol{y}})$$

on the nth example in order to satisfy the single constraint on the best sequence $\widetilde{\boldsymbol{y}} \in \mathscr{Y}(\boldsymbol{x}^n)$ found by (12).

Involving more than one constraint leads to k-best MIRA, where the optimization is performed over corresponding k-best constraints. We define the sequence of optimization problems inside one example in the online manner, where each of them is subject to only one of k-best constraints. Since only one constraint is considered at a time, each one can be solved analytically. Thus, we will sequentially traverse through k-best sequences, optimize the sub-problem restricted to only one of the sequences

$$\min_{\xi, \mathbf{w}_n^{j+1}} \frac{1}{2}\|\mathbf{w}_n^{j+1} - \mathbf{w}_n^j\|^2 + C\xi \tag{22}$$

$$\text{s.t. } \mathbf{w}_n^{j+1^{\mathsf{T}}} \varDelta\mathbf{F}_n(\boldsymbol{y}^{(n,j)}) \geq L(\boldsymbol{y}^n, \boldsymbol{y}^{(n,j)}) - \xi, \quad \xi \geq 0, \tag{23}$$

for $j = 1, \ldots, k$ and perform passive-aggressive updates

$$\mathbf{w}_n^{j+1} = \mathbf{w}_n^j + \delta_{n,j}\varDelta\mathbf{F}_n(\boldsymbol{y}^{(n,j)}), \quad j = 1, \ldots, k, \tag{24}$$

where each step can be found in the closed form as

$$\delta_{n,j} = \min \left\{ \frac{\max\{0, \ell(\mathbf{w}_n^j; (\boldsymbol{x}^n, \boldsymbol{y}^{(n,j)}))\}}{\|\varDelta\mathbf{F}_n(\boldsymbol{y}^{(n,j)})\|^2}, C \right\} . \tag{25}$$

With previous two formulas we define *k-best passive-aggressive updates*, supposing that we start processing the nth example with parameters $\mathbf{w}_n$, which are used to produce k-best sequences $(\boldsymbol{y}^{(n,1)}, \ldots, \boldsymbol{y}^{(n,k)}) = \mathscr{B}^k_{\mathbf{w}_n}$. The parameters $\mathbf{w}_n^j$ denote the intermediate states of parameters through the iterations between starting parameters $\mathbf{w}_n = \mathbf{w}_n^1$ and parameters after the optimization on the nth example $\mathbf{w}_{n+1} = \mathbf{w}_n^{k+1}$. While k-best MIRA can be seen as a semi-online algorithm (online through examples and batch inside one example) the k-best passive-aggressive algorithm is defined in a complete online manner through the examples and also inside one example.

Cramer et. al [7] provide a cumulative prediction loss bound for the passive-aggressive algorithm for cost-sensitive multiclass classification, which can also be applied to 1-best PA algorithm for a structured output. Here, we will provide a similar bound for the k-best case with a slightly different assumptions, where updates are additionally restricted from k-best sequences. Let's first define the *prediction sequence* $\widehat{\boldsymbol{y}}_{\mathbf{w}_n}$ found by parameters $\mathbf{w}_n$ on the nth example as

$$\widehat{\boldsymbol{y}}_{\mathbf{w}_n} = \operatorname*{arg\,max}_{\boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n)} \mathbf{w}_n^{\mathsf{T}} \mathbf{F}(\boldsymbol{x}^n, \boldsymbol{y}). \tag{26}$$

This sequence is also considered in Crammer et. al [7] for use in the PA approach, where steps corresponding to $\widehat{\boldsymbol{y}}$ and $\widetilde{\boldsymbol{y}}$ are called the prediction-based and the max-loss step, respectively. We use this sequence to define the auxiliary set

$$A_{\mathbf{w}_n} = \{\boldsymbol{y} \in \mathscr{Y}_{-n} : \ell(\mathbf{w}_n; (\boldsymbol{x}^n, \boldsymbol{y})) \geq \ell(\mathbf{w}_n; (\boldsymbol{x}^n, \widehat{\boldsymbol{y}}_{\mathbf{w}_n}))\}, \tag{27}$$

which we need to define the *restricted k-best passive-aggressive* parameter updates as

$$\mathbf{w}_n^{j+1} = \begin{cases} \mathbf{w}_n^j + \delta_{n,j} \Delta\mathbf{F}_n(\boldsymbol{y}^{(n,j)}) & \text{, if } \boldsymbol{y}^{(n,j)} \in A_{\mathbf{w}_n^j} \\ \mathbf{w}_n^j & \text{, if } \boldsymbol{y}^{(n,j)} \notin A_{\mathbf{w}_n^j} \end{cases}, \; j = 1, \ldots, k, \tag{28}$$

and use it for *k-best restricted passive aggressive (RPA) algorithm* (see Algorithm 4). Let all sequences from $\mathscr{B}^k_{\mathbf{w}_n}$ on which we make a non-zero update according to updates (28) be $S_n = (\boldsymbol{y}^{(n,i_1)}, \ldots, \boldsymbol{y}^{(n,i_{|S_n|})})$, for some indices $1 \leq i_1 < \ldots < i_{|S_n|} \leq k$ where the length of vector S_n is denoted with $|S_n|$. Further in this section, just for simplicity, we will refer to these sequences as $S_n = (\boldsymbol{y}^{(n,1)}, \ldots, \boldsymbol{y}^{(n,|S_n|)})$.

Lemma 1. *Let $(\boldsymbol{x}^1, \boldsymbol{y}^1), \ldots, (\boldsymbol{x}^N, \boldsymbol{y}^N)$ be a sequence of examples and let $\mathbf{u}$ be any parameter vector. For the restricted k-best PA update defined by (28), it follows*

$$\sum_{n=1}^{N} \sum_{j=1}^{|S_n|} \delta_{n,j} \left(2\ell(\mathbf{w}_n^j; (\boldsymbol{x}^n, \boldsymbol{y}^{(n,j)})) - \delta_{n,j} \|\Delta\mathbf{F}_n(\boldsymbol{y}^{(n,j)})\|^2 - 2\ell(\mathbf{u}; (\boldsymbol{x}^n, \boldsymbol{y}^{(n,j)}))\right) \leq \|\mathbf{u}\|^2,$$

where $S_n = (\boldsymbol{y}^{(n,1)}, \ldots, \boldsymbol{y}^{(n,|S_n|)})$ contains all sequences from $\mathscr{B}^k_{\mathbf{w}_n^1}$ on which we make a non-zero update according to (28).

Proof. Defining γ_n^j as $\|\mathbf{w}_n^j - \mathbf{u}\|^2 - \|\mathbf{w}_n^{j+1} - \mathbf{u}\|^2$ and summing it over all n and j we get

$$\begin{aligned} \sum_{n=1}^{N} \sum_{j=1}^{|S_n|} \gamma_n^j &= \sum_{n=1}^{N} \sum_{j=1}^{|S_n|} \left(\|\mathbf{w}_n^j - \mathbf{u}\|^2 - \|\mathbf{w}_n^{j+1} - \mathbf{u}\|^2\right) \\ &= \|\mathbf{w}_1^1 - \mathbf{u}\|^2 - \|\mathbf{w}_N^{|S_N|+1} - \mathbf{u}\|^2 \leq \|\mathbf{w}_1^1 - \mathbf{u}\|^2 = \|\mathbf{u}\|^2 \end{aligned} \tag{29}$$

because $\mathbf{w}_1^1 = \mathbf{0}$ and $\mathbf{w}_{n+1}^1 = \mathbf{w}_n^{|S_n|+1}$. According to the definition of γ_n^j and the parameter change (28), we get that

$$\begin{aligned}
\gamma_n^j &= \|\mathbf{w}_n^j - \mathbf{u}\|^2 - \|\mathbf{w}_n^{j+1} - \mathbf{u}\|^2 \\
&= \|\mathbf{w}_n^j - \mathbf{u}\|^2 - \|\mathbf{w}_n^j - \mathbf{u} + \delta_{n,j} \Delta\mathbf{F}_n(\boldsymbol{y}^{(n,j)})\|^2 \\
&= -2\delta_{n,j}(\mathbf{w}_n^j - \mathbf{u})^\mathsf{T} \Delta\mathbf{F}_n(\boldsymbol{y}^{(n,j)}) - \delta_{n,j}^2 \|\Delta\mathbf{F}_n(\boldsymbol{y}^{(n,j)})\|^2 \\
&= -2\delta_{n,j}\left({\mathbf{w}_n^j}^\mathsf{T} \Delta\mathbf{F}_n(\boldsymbol{y}^{(n,j)}) - L(\boldsymbol{y}^{(n,j)}, \boldsymbol{y}^n)\right) \\
&\quad + 2\delta_{n,j}\left(\mathbf{u}^\mathsf{T} \Delta\mathbf{F}_n(\boldsymbol{y}^{(n,j)}) - L(\boldsymbol{y}^{(n,j)}, \boldsymbol{y}^n)\right) - \delta_{n,j}^2 \|\Delta\mathbf{F}_n(\boldsymbol{y}^{(n,j)})\|^2 \\
&= \delta_{n,j}\left(2\ell(\mathbf{w}_n^j; (\boldsymbol{x}^n, \boldsymbol{y}^{(n,j)})) - \delta_{n,j}\|\Delta\mathbf{F}_n(\boldsymbol{y}^{(n,j)})\|^2 - 2\ell(\mathbf{u}; (\boldsymbol{x}^n, \boldsymbol{y}^{(n,j)}))\right)
\end{aligned}$$

which after summing and using (29) provides the desired inequality.

Theorem 1. *Let $(\boldsymbol{x}^1, \boldsymbol{y}^1), \ldots, (\boldsymbol{x}^N, \boldsymbol{y}^N)$ be a sequence of examples where $\|\Delta\mathbf{F}_n(\boldsymbol{y})\| \leq 1$ and $L(\boldsymbol{y}^n, \boldsymbol{y}) \leq C$ for all $\boldsymbol{y} \in S_n$, $n = 1, \ldots, N$. Then for any parameter vector $\mathbf{u}$ and the restricted k-best update defined by (28), it follows*

$$\sum_{n=1}^{N} \sum_{j=1}^{|S_n|} L(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}_{\mathbf{w}_n^j})^2 \leq \|\mathbf{u}\|^2 + 2C \sum_{n=1}^{N} \sum_{j=1}^{|S_n|} \ell(\mathbf{u}; (\boldsymbol{x}^n, \boldsymbol{y}^{(n,j)})), \tag{30}$$

where $S_n = (\boldsymbol{y}^{(n,1)}, \ldots, \boldsymbol{y}^{(n,|S_n|)})$ contains all sequences from $\mathscr{B}^k_{\mathbf{w}_n^1}$ on which we make a non-zero update according to (28).

Proof. In proof we use abbreviations $L_{n,j} = L(\boldsymbol{y}^n, \widehat{\boldsymbol{y}}_{\mathbf{w}_n^j})$, $\widehat{\boldsymbol{y}}_{n,j} = \widehat{\boldsymbol{y}}_{\mathbf{w}_n^j}$ and $\ell_{n,j} = \ell(\mathbf{w}_n^j; (\boldsymbol{x}^n, \boldsymbol{y}^{(n,j)}))$. According to the definition (26) of the prediction sequence, it follows

$${\mathbf{w}_n^j}^\mathsf{T} \mathbf{F}(\boldsymbol{x}^n, \widehat{\boldsymbol{y}}_{n,j}) \geq {\mathbf{w}_n^j}^\mathsf{T} \mathbf{F}(\boldsymbol{x}^n, \boldsymbol{y}), \quad \forall \boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n)$$

which leads to inequality

$$L_{n,j} \leq {\mathbf{w}_n^j}^\mathsf{T} \mathbf{F}(\boldsymbol{x}^n, \widehat{\boldsymbol{y}}_{n,j}) - {\mathbf{w}_n^j}^\mathsf{T} \mathbf{F}(\boldsymbol{x}^n, \boldsymbol{y}^n) + L_{n,j} = \ell(\mathbf{w}_n^j; (\boldsymbol{x}^n, \widehat{\boldsymbol{y}}_{n,j})) \leq \ell_{n,j} \tag{31}$$

where the last inequality in (31) comes from the definition of the set S_n, i.e. because of $\boldsymbol{y}^{(n,j)} \in A_{\mathbf{w}_n^j}$. For each non-zero step $\delta_{n,j}$ it follows that $\ell_{n,j} > 0$ and according to the condition $\|\Delta\mathbf{F}_n(\boldsymbol{y}^{(n,j)})\| \leq 1$, $j = 1, \ldots, |S_n|$, we get

$$\delta_{n,j} = \min\left\{\frac{\max\{0, \ell_{n,j}\}}{\|\Delta\mathbf{F}_n(\boldsymbol{y}^{(n,j)})\|^2}, C\right\} \geq \min\{\ell_{n,j}, C\}$$

which leads to

$$\delta_{n,j} L_{n,j} \geq \min\{\ell_{n,j} L_{n,j}, C L_{n,j}\} \overset{(31)}{\geq} \min\{L_{n,j}^2, C L_{n,j}\} \geq L_{n,j}^2 \tag{32}$$

since $L_{n,j} \leq C$. Summing (32) for all examples (n, j) on which we made the non-zero update and using the inequality from (31) we get

$$\sum_{n=1}^{N} \sum_{j=1}^{|S_n|} L_{n,j}^2 \leq \sum_{n=1}^{N} \sum_{j=1}^{|S_n|} \delta_{n,j} L_{n,j} \leq \sum_{n=1}^{N} \sum_{j=1}^{|S_n|} \delta_{n,j} \ell_{n,j}. \tag{33}$$

According to the definition of step size $\delta_{n,j}$, it is upper bound by the parameter C, which allows us to rewrite the inequality from Lemma 1 as

$$\sum_{n=1}^{N}\sum_{j=1}^{|S_n|}\delta_{n,j}\Big(2\ell_{n,j}-\delta_{n,j}\|\Delta\mathbf{F}_n(\boldsymbol{y}^{(n,j)})\|^2\Big)\leq\|\mathbf{u}\|^2+2C\sum_{n=1}^{N}\sum_{j=1}^{|S_n|}\ell(\mathbf{u};(\boldsymbol{x}^n,\boldsymbol{y}^{(n,j)})).$$

Also, the definition of $\delta_{n,j}$ for every non zero step gives $\delta_{n,j}\|\Delta\mathbf{F}_n(\boldsymbol{y}^{(n,j)})\|^2\leq\ell_{n,j}$ and thus the previous inequality becomes

$$\sum_{n=1}^{N}\sum_{j=1}^{|S_n|}\delta_{n,j}\ell_{n,j}\leq\|\mathbf{u}\|^2+2C\sum_{n=1}^{N}\sum_{j=1}^{|S_n|}\ell(\mathbf{u};(\boldsymbol{x}^n,\boldsymbol{y}^{(n,j)})),\tag{34}$$

which in combination with (33) provides the desired bound.

Corollary 1. *Let the conditions from the previous theorem be satisfied, then it follows*

$$\sum_{n=1}^{N}\sum_{j=1}^{|S_n|}L(\boldsymbol{y}^n,\widehat{\boldsymbol{y}}_{\mathbf{w}_n^j})^2\leq\|\mathbf{u}\|^2+2C\sum_{n=1}^{N}|S_n|\ell_n(\mathbf{u}).\tag{35}$$

where $\ell_n(\mathbf{u})=\max_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)}\ell(\mathbf{u};(\boldsymbol{x}^n,\boldsymbol{y}))$.

In case of using only the best sequence, i.e. $S_n=(\widetilde{\boldsymbol{y}})$, the bound from Corollary 1 reduces to the bound for 1-best case proved by Crammer et al. [7]. The proof of 1-best case uses a property $\ell(\mathbf{w}_n;(\boldsymbol{x}^n,\widetilde{\boldsymbol{y}}))\geq\ell(\mathbf{w}_n;(\boldsymbol{x}^n,\widehat{\boldsymbol{y}}))$ which is needed for inequality (32). However, this inequality does not hold if we change $\widetilde{\boldsymbol{y}}$ with an arbitrary sequence from k-best ones. In order to get a similar bound, updates must be restricted only to those examples which belong to the set $A_{\mathbf{w}_n}$. Nevertheless, checking if a sentence belongs to this set implies finding $\widehat{\boldsymbol{y}}$ every time we change the parameters (see Algorithm 4) which is computationally expensive. In the experiment section, we will consider both restricted k-best PA and k-best PA updates, even though for latter the previously proved prediction loss bound will not be satisfied.

4.5. *k*-best Perceptron

In case of not using a cost function, i.e. when $L(\boldsymbol{y}^n,\boldsymbol{y})=0$ for all $\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)$, the passive-aggressive algorithm reduces to perceptron algorithm [17] with the fixed step size for the best sequence. To keep the spirit of the online manner of the single best perceptron, the k-best version should involve online traversing through k-best structures $(\boldsymbol{y}^{(n,1)},\ldots,\boldsymbol{y}^{(n,k)})=\mathscr{B}^k_{\mathbf{w}_n}$ and sequential changes of parameters with the constant step size for each structure which belongs to the error set, i.e. for each structure which produces a higher score than the original structure. After finding the k-best structure, the following series of parameter change is applied

$$\mathbf{w}_n^{j+1}=\begin{cases}\mathbf{w}_n^j+\Delta\mathbf{F}_n(\boldsymbol{y}^{(n,j)}) & \text{, if } \boldsymbol{y}^{(n,j)}\in Err_{\mathbf{w}_n^j}\\ \mathbf{w}_n^j & \text{, if } \boldsymbol{y}^{(n,j)}\notin Err_{\mathbf{w}_n^j}\end{cases},\quad j=1,\ldots,k\tag{36}$$

where $Err_{\mathbf{w}_n}=\{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n):\mathbf{w}_n^{\mathsf{T}}\Delta\mathbf{F}_n(\boldsymbol{y})\leq 0\}$. The condition under which the k-best perceptron converges is the same as for 1-best case.

Theorem 2. *Let $\mathscr{D} = ((\boldsymbol{x}^n, \boldsymbol{y}^n))_{n=1}^N$ be a training set and let's suppose that there exist a vector $\mathbf{u}$ and $\gamma > 0$ such that $\|\mathbf{u}\| = 1$ and $\mathbf{u}^\mathsf{T} \Delta\mathbf{F}_n(\boldsymbol{y}) \geq \gamma$ for all training examples n and for all $\boldsymbol{y} \in \mathscr{Y}_{-n}$. For the k-best perceptron from Algorithm 4, it follows that*

$$\textit{Number of mistakes} \leq \frac{R^2}{\gamma^2}$$

where R is a constant such that $\forall n, \forall \boldsymbol{y} \in \mathscr{Y}_{-n}$, $\|\Delta\mathbf{F}_n(\boldsymbol{y})\| < R$.

The proof is given in the Appendix and it is very similar to the standard structured perceptron [17] since the property of the best sequence is not used in the proof.

In the case of inseparable data, there is also a theorem that bounds the number of mistakes, proven in [18] for the online perceptron and extended in [17] for the structured perceptron. The same bound holds for k-best variant.

Theorem 3. *Let $\mathscr{D} = ((\boldsymbol{x}^n, \boldsymbol{y}^n))_{n=1}^N$ be a training set, let $\mathbf{u}$ be any vector with $\|\mathbf{u}\| = 1$ and $\gamma > 0$. Define*

$$m_n = \mathbf{u}^\mathsf{T}\mathbf{F}(\boldsymbol{x}^n, \boldsymbol{y}^n) - \max_{\boldsymbol{y} \in \mathscr{Y}_{-n}} \mathbf{u}^\mathsf{T}\mathbf{F}(\boldsymbol{x}^n, \boldsymbol{y}), \quad \epsilon_n = \max\{0, \gamma - m_n\}$$

and $D = \sqrt{\sum_{n=1}^N \epsilon_n^2}$. For the first pass over the training set of k-best perceptron from Algorithm 4,

$$\textit{Number of mistakes} \leq \left(\frac{R + D}{\gamma}\right),$$

where R is a constant such that $\forall n, \forall \boldsymbol{y} \in \mathscr{Y}_{-n}$, $\|\Delta\mathbf{F}_n(\boldsymbol{y})\| < R$.

The idea of the proof is to transform an inseparable case into a separable one, then apply the theorem for a separable case to get the bound, and at the end show that the prediction with the original parameters is the same as with the transformed parameters. The proof is identical as in [17], where the only difference is that we apply Theorem 2 for k-best perceptron when we get a separable case. For completeness the proof is given in the Appendix.

5. Results and discussion

We present experimental results on two sequence labeling tasks, shallow parsing [6] on CONLL-2000 corpus[2] and named entity recognition in Spanish on CONLL-2002 corpus[3] [27] and in English on MUC-7 corpus[4] [28]. In further text, we will address algorithms by their names removing the "k-best" prefix, and where needed, specifying the exact parameter.

[2] http://www.cnts.ua.ac.be/conll2000/chunking

[3] http://www.cnts.ua.ac.be/conll2002/ner

[4] https://catalog.ldc.upenn.edu/LDC2001T02

Algorithm 4: k-best perceptron, k-best PA and k-best restricted PA (RPA)

```
Input  : Training data: $\mathscr{D} = ((\boldsymbol{x}^n, \boldsymbol{y}^n))_{n=1}^{N}$, parameter $C \in \mathbb{R}^+$, $k \in \mathbb{N}$
         Number of iterations: $P$
Output: Model parameters: $\mathbf{w}$

$\mathbf{w} \leftarrow \mathbf{0}$;
for $p \leftarrow 1$ to $P$ do
    foreach $(\boldsymbol{x}^n, \boldsymbol{y}^n)$ in $\mathscr{D}$ do
        $S \leftarrow \mathscr{B}_{\mathbf{w}}^{k}$; update $\leftarrow$ true;
        foreach $\boldsymbol{y}$ in $S$ do
            $Err \leftarrow \begin{cases} \mathbf{w}^{\mathsf{T}} \Delta\mathbf{F}_n(\boldsymbol{y}) < L(\boldsymbol{y}^n, \boldsymbol{y}) & \text{(PA, RPA)} \\ \mathbf{w}^{\mathsf{T}} \Delta\mathbf{F}_n(\boldsymbol{y}) \leq 0 & \text{(Perceptron)} \end{cases}$
            if update then
                $\widehat{\boldsymbol{y}} \leftarrow \arg\max_{\boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n)} \mathbf{w}^{\mathsf{T}} \mathbf{F}(\boldsymbol{x}^n, \boldsymbol{y})$;        (RPA)
            $Err \leftarrow Err$ and $\ell(\mathbf{w}; (\boldsymbol{x}^n, \boldsymbol{y})) \geq \ell(\mathbf{w}; (\boldsymbol{x}^n, \widehat{\boldsymbol{y}}))$;    (RPA)
            if $Err$ then
                $\delta \leftarrow \begin{cases} \min\left\{\ell(\mathbf{w}; (\boldsymbol{x}^n, \boldsymbol{y})) / \left\|\Delta\mathbf{F}_n(\boldsymbol{y})\right\|^2, C\right\} & \text{(PA, RPA)} \\ 1 & \text{(Perceptron)} \end{cases}$
                $\mathbf{w} \leftarrow \mathbf{w} + \delta\, \Delta\mathbf{F}_n(\boldsymbol{y})$; update $\leftarrow$ true;
            else
                update $\leftarrow$ false;
```

5.1. Problem description and features

Shallow parsing or *chunking* is a task of identifying non-overlapping text segments which correspond to certain syntactic units (chunks), such as a noun phrase, verb phrase, prepositional phrase, etc. The CONLL-2000 corpus contains around a quarter of a million words already split for training and testing. Each word has a corresponding POS tag and a label. The labels are presented in BIO representation, where B stands for the beginning of a chunk, I for the interior, and O means that a word does not belong to any chunk. For each word we first detect its characteristics which we use as a local feature. We extract standard features like the detection of special characters, the detection of numbers, the characteristic suffix of the word, belonging to a characteristic dictionary, whether the word is capitalized or all caps. All bigrams are constructed for a word and its local feature (including the POS tag) for the current and previous position, while unigrams are constructed only for the current position. These bigrams and unigrams with the combination of the current and previous label are used to create a feature vector at the current position. The results are presented in terms of F-measure, as a harmonic mean of a precision and recall computed over tokens belonging to a chunk.

Named entity recognition (NER) is a task of detecting and classifying entities into specified categories, such as names of persons, organizations, locations, times, dates, etc. For NER in Spanish, the CoNLL-2002 corpus is divided into the training, test and development part containing four types of entities: person, organization, location and miscellaneous, while for NER in English, the corpus has additional four entity types: date, time, money and percent, instead of the miscellaneous type. The feature vectors are created in

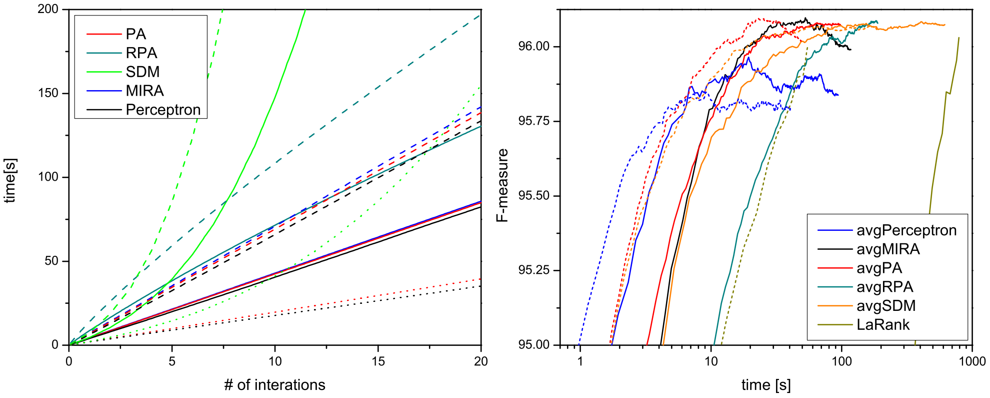

Fig. 2. Training time comparison for different k-best algorithms on shallow parsing. The left panel shows time through iterations for different values of the parameter k, where the algorithms are denoted with colors, while k is represented with a line style: $k = 1$ with dots, $k = 5$ with a solid line, and $k = 10$ with a dashed line. The right panel shows the F-measure through time for $k = 1$ (dashed line) and $k = 5$ (solid line), where the LaRank trained in one pass uses parameter $n_R = 10$. In the supplementary material, this figure is broken into several plots for better clarity.

the same way as in the shallow parsing problem. For NER in Spanish, the algorithms are evaluated without any additional external knowledge of the language, while for English we used the same word characteristics for a local feature as in the shallow parsing problem.

5.2. Time and accuracy comparison

We have implemented all described algorithms in C++. The experiments are performed on a computer with Intel Core 2 Duo CPU 2.33 GHz and 8 GB RAM. We use A* decoding with Viterbi scores for the heuristic function to find the k-best paths [10, 12]. To avoid oscillations during the learning process, we have applied parameter averaging as described in [17]. Such algorithms we will denote using the prefix *avg*. Fig. 2 shows the speed comparison of the considered algorithms. All implemented algorithms share the same structures and operators when working with feature and weight vectors, thus the speed comparison shown in Fig. 2 can be considered reliable. The perceptron as the simplest method is slightly faster than the PA algorithm but on the right panel we can see that the other algorithm which incorporates the cost function provides better results. Recall that the RPA algorithm needs an additional 1-best decoding, and as a result of the necessity for additional decoding, for each k, its time deviates from the other algorithms with similar time consumption, the PA, perceptron and MIRA. The SDM requires significantly more time through iterations due to its continuously increasing active set of constraints. The heuristics from [13], which control the set growth, can help reducing the training time.

Next, we tested the LaRank algorithm trained in one pass. The results are presented in Fig. 3 for different values of parameters k and n_R. Since the selection of examples in REPROCESS operations is subject to random function, we presented the mean value of F-measure with the corresponding standard deviation. In order to select regularization parameter C for the shallow parsing problem and NER in English, we perform 5-fold

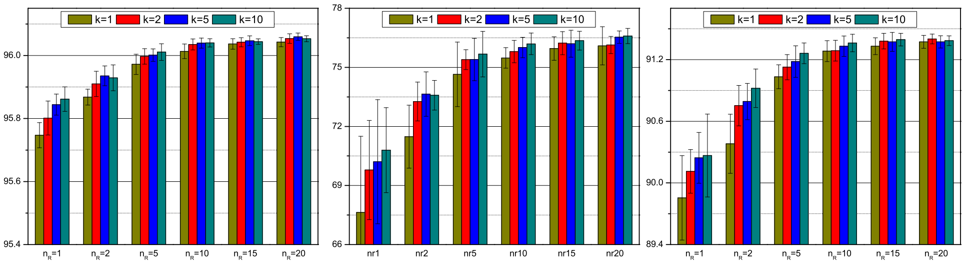

Fig. 3. The results for k-best LaRank on shallow parsing (left), NER in Spanish (middle) and NER in English (right), for different k specified in legend and for different values of parameter n_R. Each column height represents the mean value of F-measure over 20 repetitions and the corresponding error bar represents the standard deviation.

cross-validation. We select the highest mean value of F-measure over 20 repetitions, and then we use this optimal parameter in a test scenario. For NER in Spanish, we use the development set to select the optimal parameter with the same scenario. The results on all corpora suggest the advantage of k-best versions over the single best version, especially with the lower values of parameter n_R. Also, we can see that a higher number of n_R has a positive influence to the F-measure.

Further, for different values of parameter k we present a single number for each algorithm where the other parameters (regularization parameter, number of training iterations) are 5-fold cross-validated on the shallow parsing problem and NER in English, and estimated on the development set for NER in Spanish. For shallow parsing we select the best combination of the regularization parameter $C \in \{10^{-2}, 10^{-1}, 1, 10\}$ and the number of training iterations from set $\{5, 10, 15, 20\}$. For that problem, algorithms require less iterations to converge, while 30 iterations are also added to the previous set for NER, as the results were still improving after 20 iterations. Results are presented in Table 1. MIRA was optimized with the SMO with the practical check of KKT conditions (17)-(18) by setting tolerance $\tau = 10^{-10}$ for all values k and thus for $k = 1$ its results do not match the PA algorithm. For $k = 10$ there is usually a degradation of results, possibly because the inclusion of a lot of features into a training procedure via k-best sequences can raise the problem of overfitting. Another problem with the higher k can rise in algorithms such as the PA, RPA and perceptron algorithm which are defined in an online manner inside an example, as opposite to MIRA and SDM which perform full optimization inside an example. However, we can see that all k-best versions of algorithms make an improvement over the single best case and the best results are usually achieved with smaller values $k = 2$ and $k = 5$. We tested the statistical significance of these improvements of the results by running McNemar's test [30] on all datasets. With the confidence level 0.05 the improvements of the k-best version over the single best one are significant for the Perceptron, (R)PA algorithm and MIRA, while for the SDM they are not significant (more details about this test in the supplementary file).

Shallow parsing												
	$k=1$			$k=2$			$k=5$			$k=10$		
Method	C	#	F-measure	C	#	F-measure	C	#	F-measure	C	#	F-measure
Perc.	–	5	95.821	–	5	95.892	–	5	**95.924**	–	5	95.875
PA	1	10	96.093	10^{-2}	15	**96.097**	10^{-2}	15	96.069	10^{-2}	15	96.056
RPA	1	10	96.093	10^{-2}	20	**96.099**	10^{-2}	20	96.079	10^{-2}	15	96.057
MIRA	1	10	96.066	10^{-1}	10	96.053	10^{-1}	5	**96.071**	10^{-2}	20	96.061
SDM	10^{-1}	20	96.057	10^{-1}	20	96.080	10^{-1}	20	96.075	10^{-1}	20	**96.081**

Named entity recognition (Spanish)												
	$k=1$			$k=2$			$k=5$			$k=10$		
Method	C	#	F-measure	C	#	F-measure	C	#	F-measure	C	#	F-measure
Perc.	–	30	75.886	–	30	76.019	–	30	**76.204**	–	30	76.122
PA	10^{-1}	30	76.349	10^{-1}	30	76.436	1	30	**76.640**	10^{-1}	30	76.549
RPA	10^{-1}	30	76.349	1	30	76.636	10^{-1}	30	**76.741**	10^{-1}	30	76.608
MIRA	1	30	76.262	1	30	**76.334**	1	30	76.312	10^{-1}	30	76.328
SDM	1	30	75.840	1	30	76.028	1	30	**76.194**	1	30	76.145

Named entity recognition (English)												
	$k=1$			$k=2$			$k=5$			$k=10$		
Method	C	#	F-measure	C	#	F-measure	C	#	F-measure	C	#	F-measure
Perc.	–	20	90.918	–	15	90.966	–	10	**91.136**	–	20	91.122
PA	1	30	91.307	1	10	**91.350**	10^{-1}	10	91.247	10^{-1}	10	91.293
RPA	1	30	91.307	1	10	91.343	1	10	**91.357**	10^{-1}	10	91.244
MIRA	1	10	91.324	10^{-2}	30	91.393	10^{-1}	5	**91.447**	10^{-2}	30	91.399
SDM	10^{-1}	30	91.377	10	15	**91.487**	10	10	91.415	10	10	91.435

Table 1. Results for different algorithms and their corresponding parameters (regularization parameters, parameter k, and the number of training epochs) obtained from 5-fold cross-validation (shallow parsing and NER in English) and from the development set (NER in Spanish). For all algorithms, the results are presented with averaged parameters, so the *avg* prefix is omitted in the algorithm name. The number of training iterations is denoted with # and the tolerance for KKT conditions τ is set to 10^{-10}.

6. Conclusion

In this paper, we have presented four new k-best extensions of structural max-margin classifiers. Unlike the existing k-best extension of MIRA, the proposed k-best passive-aggressive (PA), k-best restricted passive aggressive (RPA) and k-best perceptron algorithm are completely defined in an online manner, through examples and inside each example as well. These algorithms perform well with small values of k on the presented problems. They are easy to implement and, except for the RPA algorithm, very fast and suitable for large scale problems. The k-best RPA is presented in order to satisfy a cumulative loss bound similar to the one in the single best PA algorithm. On the other hand, the k-best SDM performs full optimization inside each example where it collects support vectors though iterations, making the algorithm highly computationally consuming. Even though it remembers support vectors from previous epochs, it does not always achieve better results than an online algorithm like MIRA. Finally, the extension of LaRank to the k-best case provides notable improvements in comparison to the single best case and the algorithm is suitable for training in one pass through the data.

Appendix

(Proof of Theorem 2). For a sequence $\boldsymbol{y} \in \mathscr{Y}_{-n}$ a mistake is made with parameters $\mathbf{w}$ if $\mathbf{w}^{\mathsf{T}} \Delta\mathbf{F}_n(\boldsymbol{y}) \leq 0$. Let $\overline{\mathbf{w}}^{(l)}$ be a vector before the l-th mistake. Suppose that the mistake is made on the n-th example, on the j-th sequence taken from k-best sequences generated from parameters $\mathbf{w}_n$, i.e. on the sequence $\boldsymbol{y}^{(n,j)} \in \mathscr{B}^k_{\mathbf{w}_n}$ where $\overline{\mathbf{w}}^{(l)} = \mathbf{w}^j_n$. According to the algorithm, it follows that $\overline{\mathbf{w}}^{(l+1)} = \overline{\mathbf{w}}^{(l)} + \Delta\mathbf{F}_n(\boldsymbol{y}^{(n,j)})$ and taking the inner product of both sides with parameters $\mathbf{u}$ gives

$$\mathbf{u}^{\mathsf{T}}\overline{\mathbf{w}}^{(l+1)} = \mathbf{u}^{\mathsf{T}}\overline{\mathbf{w}}^{(l)} + \mathbf{u}^{\mathsf{T}}\Delta\mathbf{F}_n(\boldsymbol{y}^{(n,j)}) \geq \mathbf{u}^{\mathsf{T}}\overline{\mathbf{w}}^{(l)} + \gamma.$$

Since $\overline{\mathbf{w}}^{(1)} = 0$ and $\mathbf{u}^{\mathsf{T}}\overline{\mathbf{w}}^{(1)} = 0$, it follows by induction that $\mathbf{u}^{\mathsf{T}}\overline{\mathbf{w}}^{(l+1)} \geq l\gamma$, and using $\mathbf{u}^{\mathsf{T}}\overline{\mathbf{w}}^{(l+1)} \leq \|\mathbf{u}\|\,\|\overline{\mathbf{w}}^{(l+1)}\|$ gives us $\|\overline{\mathbf{w}}^{(l+1)}\| \geq l\gamma$. Further

$$\|\overline{\mathbf{w}}^{(l+1)}\|^2 = \|\overline{\mathbf{w}}^{(l)}\|^2 + 2\,\overline{\mathbf{w}}^{(l)\mathsf{T}}\Delta\mathbf{F}_n(\boldsymbol{y}^{(n,j)}) + \|\Delta\mathbf{F}_n(\boldsymbol{y}^{(n,j)})\|^2 \leq \|\overline{\mathbf{w}}^{(l)}\|^2 + R^2$$

because parameters $\overline{\mathbf{w}}^{(l)}$ make a mistake on $\boldsymbol{y}^{(n,j)}$, i.e. $\overline{\mathbf{w}}^{(l)\mathsf{T}}\Delta\mathbf{F}_n(\boldsymbol{y}^{(n,j)}) \leq 0$. By induction, we get $\|\overline{\mathbf{w}}^{(l+1)}\|^2 \leq lR^2$. Combining the bounds $\|\overline{\mathbf{w}}^{(l+1)}\| \geq l\gamma$ and $\|\overline{\mathbf{w}}^{(l+1)}\|^2 \leq lR^2$ we get the upper bound for the number of mistakes

$$l^2\gamma^2 \leq \|\overline{\mathbf{w}}^{(l+1)}\|^2 \leq lR^2 \quad \Longrightarrow \quad l \leq \frac{R^2}{\gamma^2}.$$

(Proof of Theorem 3). First, we extend the feature vector $\mathbf{F}(\boldsymbol{x},\boldsymbol{y}) \in \mathbb{R}^d$ to $\overline{\mathbf{F}}(\boldsymbol{x},\boldsymbol{y}) \in \mathbb{R}^{d+N}$ such that $\overline{\mathbf{F}}_i(\boldsymbol{x},\boldsymbol{y}) = \mathbf{F}_i(\boldsymbol{x},\boldsymbol{y})$, $i = 1,\ldots,d$, and $\overline{\mathbf{F}}_{d+n}(\boldsymbol{x},\boldsymbol{y})$ is equal to Z if $(\boldsymbol{x},\boldsymbol{y}) = (\boldsymbol{x}^n,\boldsymbol{y}^n)$ and equal to zero otherwise, for $n = 1,\ldots,N$. The vector $\mathbf{u}$ is extended to $\overline{\mathbf{u}} \in \mathbb{R}^{d+N}$ in a similar way: $\overline{\mathbf{u}}_i = \mathbf{u}_i$, $i = 1,\ldots,d$ and $\overline{\mathbf{u}}_{d+n} = \epsilon_n/Z$, $n = 1,\ldots,N$. Transformed vectors hold following properties

$$\begin{gathered}\|\overline{\mathbf{u}}\|^2 = \|\mathbf{u}\|^2 + \textstyle\sum_{n=1}^N \epsilon_n^2/Z^2 = 1 + D^2/Z^2 \\ \forall n, \forall \boldsymbol{y} \in \mathscr{Y}_{-n}, \overline{\mathbf{u}}^{\mathsf{T}}\,\overline{\mathbf{F}}(\boldsymbol{x}^n,\boldsymbol{y}^n) - \overline{\mathbf{u}}^{\mathsf{T}}\,\overline{\mathbf{F}}(\boldsymbol{x}^n,\boldsymbol{y}) \geq \gamma \\ \forall n, \forall \boldsymbol{y} \in \mathscr{Y}_{-n}, \|\overline{\mathbf{F}}(\boldsymbol{x}^n,\boldsymbol{y}^n) - \overline{\mathbf{F}}(\boldsymbol{x}^n,\boldsymbol{y})\| < R^2 + Z^2\end{gathered}$$

From the first two properties, it follows that parameters $\overline{\mathbf{u}}/\|\overline{\mathbf{u}}\|$ separate data $\mathscr{D}$ with the margin $\gamma/\sqrt{1 + D^2/Z^2}$. Now, we can apply Theorem 2 and get that the number of mistakes of k-best perceptron running on extended space is at most $\frac{1}{\gamma}(R^2+Z^2)(1+\frac{D^2}{Z^2})$. The value $Z = \sqrt{RD}$ minimizes the bound, giving us the statement of the theorem. Extended parameters generated from the first pass of the algorithm make the same prediction as the original parameters on test examples since the additional parameters affect only single training data.

Acknowledgments. This research was supported by Ministry of Education, Science and Technological Development, Republic of Serbia, Grant No. 174013.

References

1. Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann and Yasemin Altun, Large Margin Methods for Structured and Interdependent Output Variables, *The Journal of Machine Learning Research*, **6**, (2005), 1453–1484.
2. Koby Crammer, Ryan Mcdonald, and Fernando Pereira, Scalable large-margin online learning for structured classification, In *NIPS Workshop on Learning With Structured Outputs*, (2005).
3. Altun, Y., Tsochantaridis, I., and Hofmann, T. Hidden Markov support vector machines, In *International conference on machine learning (ICML)*, (2003), pp. 3–10
4. Thorsten Joachims, Thomas Finley, and Chun-Nam John Yu, Cutting-plane training of structural SVMs, *Machine Learning*, **77**(1), (2009), 27–59.
5. Taskar, B., Guestrin, C., and Koller, D., Max-margin Markov networks, In *Advances in Neural Information Processing Systems (NIPS)* 16, (2004).
6. Erik F. Tjong Kim Sang, Sabine Buchholz, and Kim Sang, Introduction to the CoNLL-2000 Shared Task: Chunking, In *Proc. CoNLL-2000 and LLL-2000*, (2000), pp. 127–132.
7. Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer, Online Passive-Aggressive Algorithms, *Journal of Machine Learning Research*, **7**, (2006), 551–585.
8. Ryan McDonald, Koby Crammer and Fernando Pereira, Online Large-Margin Training of Dependency Parsers, In *Proc. of the 43rd Annual Meeting on Association for Computational Linguistics (ACL)*, (2005), 91–98.
9. Lawrence R. Rabiner, A tutorial on hidden markov models and selected applications in speech recognition, In *Proceedings of the IEEE*, **77**(2), (1989), 257–286.
10. Frank K. Soong and Huang E, A Tree-Trellis Based Fast Search for Finding the N Best Sentence Hypotheses in Continuous Speech Recognition, In *Proceedings of the workshop on Speech and Natural Language*, (1990), pp. 12-19.
11. Harold W. Kuhn, and Albert W. Tucker, Nonlinear programming, In *Proceedings of 2nd Berkeley Symposium*, (1951), pp. 481-492.
12. Masa. Aki Nagata, A Stochastic Japanese Morphological Analyzer Using a Forward-DP Backward-A* N-Best Search Algorithm, in *Proc. of COLING-94*, (1994), pp. 201–207.
13. P. Balamurugan, Shirish Shevade, S. Sundararajan and S. Sathiya Keerthi, A Sequential Dual Method for Structural SVMs, In *SIAM Inter. Conf. on Data Mining*, (2011), pp. 223-234.
14. John Platt, Fast training of support vector machines using sequential minimal optimization, In *Advances in Kernel Methods: Support Vector Learning*, (MIT Press, 1999), pp. 185-208.
15. Vladimir Vapnik, Statistical Learning Theory, (Wiley, 1998).
16. Koby Crammer and Yoram Singer, Ultraconservative Online Algorithms for Multiclass Problems, *Journal of Machine Learning Research*, **3**, (2003), 951–991.
17. Michael Collins, Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms, In *Proc. of EMNLP*, vol. 10, (2002), pp. 1–8.
18. Yoav Freund and Robert E. Schapire, Large Margin Classification Using the Perceptron Algorithm, *Machine Learning*, **37**(3), (1999), 277–296.
19. Kevin Gimpel and Shay Cohen, Discriminative Online Algorithms for Sequence Labeling - A Comparative Study, (2007).
20. Dejan Mančev and Branimir Todorović, A primal sub-gradient method for structured classification with the averaged sum loss, *International Journal of Applied Mathematics and Computer Science* **24**(4), (2014), pp. 917–930
21. Antoine Bordes, Léon Bottou, Patrick Gallinari, and Jason Weston, Solving multiclass support vector machines with LaRank, In *Proc. of the 24th ICML*, (2007), pp. 89–96.
22. Antoine Bordes, Nicolas Usunier and Léon Bottou, Sequence Labelling with SVMs Trained in One Pass, In *ECML PKDD 2008, Part I, LNAI 5211*, (2008), pp. 146-161.
23. Ben Taskar, Learning Structured Prediction Models: A Large Margin Approach, *Doctoral dissertation*, (Stanford University, 2004).

24. Andrew McCallum, Dayne Freitag, and F. C. N. Pereira, Maximum Entropy Markov Models for Information Extraction and Segmentation, In *Proc. of the 17th ICML*, (2000), pp. 591-598.
25. John D. Lafferty, Andrew McCallum, and F. C. N. Pereira, Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data, In *ICML*, (2001), pp. 282–289.
26. Antoine Bordes, Seyda Ertekin, Jason Weston, and Léon Bottou, Fast Kernel Classifiers with Online and Active Learning, *Journal of Machine Learning Research*, **6**, (2005), 1579–1619.
27. Erik F. Tjong Kim Sang, Introduction to the CoNLL-2002 Shared Task: Language-Independent Named Entity Recognition, In *Proc. of CoNLL-2002*, (2002), pp. 155-158.
28. Nancy Chinchor and Patricia Robinson, MUC-7 named entity task definition. *In Proceedings of the 7th Conference on Message Understanding*, (1997).
29. Steven L. Salzberg, On Comparing Classifiers: Pitfalls to Avoid and a Recommended Approach, *Data mining and knowledge discovery* **1**, (1997), 317-328.
30. Quinn McNemar, Note on the sampling error of the difference between correlated proportions or percentages, *Psychometrika* **12**, (1947), 153–157.
31. Martin Jaggi, Simon Lacoste-Julien, Mark Schmidt and Patrick Pletscher, Block-Coordinate Frank-Wolfe Optimization for Structural SVMs, In *Proc. of the 30th ICML*, (2013), 53-61.
32. Avihai Mejer and Koby Crammer, Confidence in structured-prediction using confidence-weighted models, In *Proc. of the Conf. on Empirical Methods in NLP*, (2010), pp. 971–981.
33. Dejan Mančev and Branimir Todorović, Confidence based learning of a two-model committee for sequence labeling, In *Proc. of NEUREL*, (2012), pp. 167-170.
34. Shai Shalev-Shwartz, Yoram Singer, Nathan Srebro, and Andrew Cotter, Pegasos: primal estimated sub-gradient solver for SVM. *Mathematical Programming*, Vol. **127**, (2011), pp. 3–30
35. Ming-Wei Chang and Wen-tau Yih, Dual Coordinate Descent Algorithms for Efficient Large Margin Structured Prediction, *Trans. of the Assoc. for Comp. Ling.* **1**, (2013), 207–218.

Dejan Mančev is currently a teaching assistant and a Ph.D. student at the Faculty of Science and Mathematics, University of Niš, Serbia. He received a M.Sc degree in Mathematics in 2008 from the same university. His research interests include training methods for structured classifiers with application in natural language processing and the application of neural networks in the stock market forecast.

Branimir Todorović is associate professor at Computer Science Department, Faculty of Mathematics and Sciences, University of Niš. He received his Doctor of Science degree from Faculty of Electrical Engineering, University of Belgrade. His research interest include sequential Bayesian training of feed forward and recurrent neural networks, blind source separation and deconvolution, on line training of structural classifiers, active and semi-supervised learning algorithms.

Received: July 13, 2014; Accepted: April 30, 2015