Low-Dimensional Discriminative Reranking

Jagadeesh Jagarlamudi
Department of Computer Science
University of Maryland
College Park, MD 20742, USA
jags@umiacs.umd.edu

Hal Daumé III
Department of Computer Science
University of Maryland
College Park, MD 20742, USA
hal@umiacs.umd.edu

Abstract

The accuracy of many natural language processing tasks can be improved by a reranking step, which involves selecting a single output from a list of candidate outputs generated by a baseline system. We propose a novel family of reranking algorithms based on learning *separate* low-dimensional embeddings of the task's input and output spaces. This embedding is learned in such a way that prediction becomes a low-dimensional nearest-neighbor search, which can be done computationally efficiently. A key quality of our approach is that feature engineering can be done *separately* on the input and output spaces; the relationship between inputs and outputs is learned automatically. Experiments on part-of-speech tagging task in four languages show significant improvements over a baseline decoder and existing reranking approaches.

1 Introduction

Mapping inputs to outputs lies at the heart of many Natural Language Processing applications. For example, given a sentence as input: part-of-speech (POS) tagging involves finding the appropriate POS tag sequence (Thede and Harper, 1999); parsing involves finding the appropriate tree structure (Kubler et al., 2009) and statistical machine translation (SMT) involves finding correct target language translation (Brown et al., 1993). The accuracy achieved on such tasks can often be improved significantly with the help of a discriminative reranking step (Collins and Koo, 2005; Charniak and Johnson, 2005; Shen et al., 2004; Watanabe et al., 2007). For the POS tagging, reranking is relative less explored due to the already higher accuracies in English (Collins, 2002), but it is shown to improve accuracies in other languages such as Chinese (Huang et al., 2007). In this paper, we propose a novel approach to discriminative reranking and show its effectiveness in POS tagging. Reranking allows us to use arbitrary features defined jointly on input and output spaces that are often difficult to incorporate into the baseline decoder due to the computational tractability issues. The effectiveness of reranking depends on the joint features defined over both input and output spaces. This has led the community to spend substantial efforts in defining joint features for reranking (Fraser et al., 2009; Chiang et al., 2009).

Unfortunately, developing joint features over the input and output space can be challenging, especially in problems for which the exact mapping between the input and the output is unclear (for instance, in automatic caption generation for images, semantic parsing or non-literal translation). In contrast to prior work, our approach uses features defined *separately* within the input and output spaces, and learns a mapping function that can map an object from one space into the other. Since our approach requires within-space features, it makes the feature engineering relatively easy.

For clarity, we will discuss our approach in the context of POS tagging, though of course it generalizes to any reranking problem. At test time, in POS tagging, we receive a sentence and a list of candidate output POS sequences as input. We run a feature extractor on the input sentence to obtain a representation $\mathbf{x} \in \mathbb{R}^{d_1}$; we run an *independent* feature extractor on each of the m-many outputs to obtain representations $\hat{\mathbf{y}}_1, \ldots, \hat{\mathbf{y}}_m \in \mathbb{R}^{d_2}$. We will *project* all of these points down to a low k-dimensional space by means of matrices $A \in \mathbb{R}^{d_1 \times k}$ (for $\mathbf{x}$ as $A^T\mathbf{x}$) and $B \in \mathbb{R}^{d_2 \times k}$ (for $\hat{\mathbf{y}}$ as $B^T\hat{\mathbf{y}}$). We then select as the output the $\hat{\mathbf{y}}_j$ that maximizes cosine similar to $\mathbf{x}$ in the lower-dimensional space: $\max_j cos(A^T\mathbf{x}, B^T\hat{\mathbf{y}}_j)$. The goal is to learn the projection matrices A and B so that the result of this operation is a low-loss output.

Given training data of sentences and their reference tag sequences, our approach implicitly uses all possible pairwise feature combinations across the views and learns the matrices A and B that can map a given sentence (as its feature vector) to its corresponding tag sequence. Considering all possible pairwise combinations enables our model to automatically handle long range dependencies such as a word at a position effecting the tag choice at any other position.

Experiments performed on four languages (English, Chinese, French and Swedish) show the effectiveness of our approach in comparison to the baseline decoder and to the existing reranking approaches (Sec. 4). Using only the within-space features, our models are able to beat reranking approaches that use more informative joint features. While it is possible to include joint features into our models, we leave this for future work.

2 Models for Low-Dimensional Reranking

In this section, we describe our approach to learning low-dimensional representations for reranking. We first fix some notation, then discuss the intuition behind the problem we wish to solve. We propose both generative-style and discriminative-style approaches to formalizing this intuition, as well as a softened variant of the discriminative model. In the subsequent section, we discuss computational issues related to these models.

2.1 Notation

Let $\mathbf{x}_i \in \mathbb{R}^{d_1}$ and $\mathbf{y}_i \in \mathbb{R}^{d_2}$ be the feature vectors representing the $i^{th}(1 \cdots n)$ sentence and its reference tag sequence from the training data. Each sentence is also associated with m_i number of candidate tag sequences, output by the baseline decoder, and are represented as $\hat{\mathbf{y}}_{ij} \in \mathbb{R}^{d_2}$ $j = 1 \cdots m_i$. Each candidate tag sequence ($\hat{\mathbf{y}}_{ij}$) is also associated with a non-negative loss L_{ij}. Note that we place absolutely no constraints on the loss function. Moreover, let X $(d_1 \times n)$ and Y $(d_2 \times n)$ denote the data matrices with $\mathbf{x}_i$ and $\mathbf{y}_i$ as columns respectively. Finally, let $\langle \mathbf{u}, \mathbf{v} \rangle$ denote the dot product of the two vectors $\mathbf{u}$ and $\mathbf{v}$.

2.2 Intuition

As stated in the introduction, our goal is to learn projections $A \in \mathbb{R}^{d_1 \times k}$ and $B \in \mathbb{R}^{d_2 \times k}$ in such a way that test-time predictions are made with high accuracy (or low loss). At test time, the output will be chosen by maximizing cosine similarity between the input and the output, after projecting these vectors into a low-dimensional space using A and B, respectively. The cosine similarity in our context is:

$$\frac{\mathbf{x}^T A B^T \hat{\mathbf{y}}_j}{\sqrt{\mathbf{x}^T A A^T \mathbf{x}} \sqrt{\hat{\mathbf{y}}_j^T B B^T \hat{\mathbf{y}}_j}} \tag{1}$$

Our goal is to learn A and B in such a way that the $\hat{\mathbf{y}}_j$ with maximum cosine similarity to an $\mathbf{x}$ is actually the correct output. In what follows, we will describe our models to find one-dimensional projection vectors $\mathbf{a} \in \mathbb{R}^{d_1}$ and $\mathbf{b} \in \mathbb{R}^{d_2}$, but the generalization to matrices A and B is very trivial.

2.3 A Generative-Style Model

The first model we propose is akin to a generative probabilistic model, in the sense that it attempts to model the relationship between an input and its desired output, without taking alternate possible outputs into account. In the context of the intuition sketched in the previous section, the idea is to choose A and B so as to maximize the cosine similarities on the training data between each input and it's correct (or minimal-loss) output. This model intentionally *ignores* the information present in the alternative, incorrect outputs. The hope is that by making the cosine similarities with the best output as high as possible, all the alternate outputs will look bad in comparison.

Given a training data of sentences and their reference tag sequences represented as X and Y (Sec. 2.1), our generative model finds projection directions, in word and tag spaces, along which the aligned sentence and tag sequence pairs have maximum cosine similarity. In the one-dimensional setting, it finds directions $\mathbf{a} \in \mathbb{R}^{d_1}$ and $\mathbf{b} \in \mathbb{R}^{d_2}$ such that the correlation as defined in Eq. 2 is maximized.

$$\frac{\mathbf{a}^T XY^T\mathbf{b}}{\sqrt{\mathbf{a}^T XX^T\mathbf{a}}\sqrt{\mathbf{b}^T YY^T\mathbf{b}}} \tag{2}$$

Since the objective is invariant to the scaling of vectors $\mathbf{a}$ and $\mathbf{b}$, it can be rewritten as:

$$\arg\max_{\mathbf{a},\mathbf{b}} \mathbf{a}^T XY^T\mathbf{b} \tag{3}$$

$$\text{s.t. } \mathbf{a}^T XX^T\mathbf{a} = 1 \quad \text{and} \quad \mathbf{b}^T YY^T\mathbf{b} = 1 \tag{4}$$

We refer to the constraints in Eq. 4 as length constraints in the rest of this paper.

To understand why maximizing this objective function learns a good mapping function between the sentence and the tag sequence, consider decomposing the objective function as follows:

$$\begin{aligned}
\mathbf{a}^T XY^T\mathbf{b} &= \sum_{i=1}^{n}\langle\mathbf{x}_i,\mathbf{a}\rangle\langle\mathbf{y}_i,\mathbf{b}\rangle \\
&= \sum_{i=1}^{n}\Big(\sum_{l=1}^{d_1}\mathbf{x}_i^l a_l \cdot \sum_{m=1}^{d_2}\mathbf{y}_i^m b_m\Big) \\
&= \sum_{i=1}^{n}\Big(\sum_{l=1}^{d_1}\sum_{m=1}^{d_2}\mathbf{x}_i^l a_l\, \mathbf{y}_i^m b_m\Big) \\
&= \sum_{i=1}^{n}\Big(\sum_{l,m=1}^{d_1,d_2} w_{lm}\phi_i^{lm}\Big) \qquad (5)
\end{aligned}$$

where we replaced the scalars $\mathbf{x}_i^l\mathbf{y}_i^m$ and $a_l b_m$ with ϕ_i^{lm} and w_{lm} respectively. So finally, the objective can be expressed as $\mathbf{a}^T XY^T\mathbf{b} = \sum_i\langle\mathbf{w},\phi(\mathbf{x}_i,\mathbf{y}_i)\rangle$ where $\mathbf{w}$ is the weight vector and $\phi(\mathbf{x}_i,\mathbf{y}_i)$ is a vector of size ($d_1 \times d_2$) and is given by the Kronecker product of the two feature vectors $\mathbf{x}_i$ and $\mathbf{y}_i$.

In this form, the generative objective function bears similarity to the linear boundary surface widely used in machine learning, except that the weights are restricted to be the outer product of two vectors. From the reduced expressions, it is clear that our generative model considers all possible pairwise combinations of the input features ($d_1 \times d_2$) and learns which of them are more important than others. Intuitively, it puts higher weight on a word and tag pair that co-occur frequently in the training data, at the same time each of these are infrequent in their own views.

2.4 A Discriminative-Style Model

The primary disadvantage of our generative model is that it only uses input sentences and their reference tag sequences and does *not* use the incorrect candidate tag sequences of a given sentence at all. In what follows, we describe a model that utilize the incorrect candidate tag sequences as negative examples to improve the projection directions ($\mathbf{a}$ and $\mathbf{b}$). Our goal is to address this by adding constraints to our model that explicitly penalize ranking high-loss outputs higher than low-loss outputs, as is often done in the context of maximum-margin structure prediction techniques (Taskar et al., 2004).

In this section, we describe a discriminative model that keeps track of the margin deviations and finds the projection directions iteratively. Intuitively, after the projection into the lower dimensional subspace, the cosine similarity of a sentence to its reference tag sequence must be greater than that of its incorrect candidate tag sequences. Moreover, the margin between these similarities should be proportional to the loss of the candidate translation, *i.e.* the more dissimilar a candidate tag sequence to its reference is, the farther it should be from the reference in the projected space.

From the decomposition shown in Eq. 5, for a given pair of source sentence $\mathbf{x}_i$ and a tag sequence $\mathbf{y}_j$, the generative model assigns a score of :

$$\langle\mathbf{a},\mathbf{x}_i\rangle\langle\mathbf{b},\mathbf{y}_j\rangle = \mathbf{a}^T\mathbf{x}_i\mathbf{y}_j^T\mathbf{b}$$

Each input sentence is also associated with a list of candidate tag sequences and since each of these candidate sequences are incorrect they should be assigned a score less than that of the reference tag sequence. Drawing ideas from structure prediction literature (Bakir et al., 2007), we modify the objective function in order to include these terms. This idea can be captured using a loss augmented margin constraint for each sentence, tag sequence pair (Tsochantaridis et al., 2004). Let ξ_i denote a non-negative slack variable, then we define our new optimization problem as:

$$\arg\max_{\mathbf{a},\mathbf{b},\xi\geq\mathbf{0}} \frac{1-\lambda}{\lambda}\mathbf{a}^T XY^T\mathbf{b} - \sum_i \xi_i \tag{6}$$

$$\text{s.t. } \mathbf{a}^T XX^T\mathbf{a} = 1 \quad \text{and} \quad \mathbf{b}^T YY^T\mathbf{b} = 1$$

$$\forall i\, \forall j \quad \mathbf{a}^T\mathbf{x}_i\mathbf{y}_i^T\mathbf{b} - \mathbf{a}^T\mathbf{x}_i\hat{\mathbf{y}}_{ij}^T\mathbf{b} \geq 1 - \frac{\xi_i}{L_{ij}}$$

where $0 \leq \lambda \leq 1$ is a weight parameter. This objective function is ensuring that the margin between the reference and the candidate tag sequences in the projected space (as given by $\mathbf{a}^T\mathbf{x}_i\mathbf{y}_i^T\mathbf{b} - \mathbf{a}^T\mathbf{x}_i\hat{\mathbf{y}}_{ij}^T\mathbf{b}$) is proportional to its loss (L_{ij}). Notice that the slack is defined for each sentence and it remains the same for all of its candidate tag sequences.

2.5 A Softened Discriminative Model

One disadvantage of the discriminative model described in the previous section is that it cannot be optimized in closed form (as discussed in the next section). In this section, we consider a model that lies between the generative model and the (fully) discriminative model. This softened model has attractive computational properties (it is easy to compute) and will also form a building block for the optimization of the full discriminative model.

For each sentence $\mathbf{x}_i$, its reference tag sequence $\mathbf{y}_i$ should be assigned a higher score than any of its candidate tag sequences $\hat{\mathbf{y}}_{ij}$ *i.e.* we want to maximize $\mathbf{a}^T\mathbf{x}_i\mathbf{y}_i^T\mathbf{b} - \mathbf{a}^T\mathbf{x}_i\hat{\mathbf{y}}_{ij}^T\mathbf{b}$. In the fully discriminative model, we enforce that this is at least one (modulo slack). In the relaxed version, we instead require that this hold *on average*. In order to achieve this we add the following terms to the objective function: $\forall j = 1 \cdots m_i$

$$\mathbf{a}^T\mathbf{x}_i\mathbf{y}_i^T\mathbf{b} - \mathbf{a}^T\mathbf{x}_i\hat{\mathbf{y}}_{ij}^T\mathbf{b} \;=\; \mathbf{a}^T\mathbf{x}_i\mathbf{r}_{ij}^T\mathbf{b} \tag{7}$$

where $\mathbf{r}_{ij} = \mathbf{y}_i - \hat{\mathbf{y}}_{ij}$ is the residual vector between the reference and the candidate sequences. Now, we simply sum all these terms for a given sentence weighted by their loss and encourage it to be as high as possible, *i.e.* we maximize

$$\frac{1}{m_i}\sum_{j=1}^{m_i} L_{ij}\Big(\mathbf{a}^T\mathbf{x}_i\mathbf{r}_{ij}^T\mathbf{b}\Big) = \mathbf{a}^T\mathbf{x}_i\Big(\frac{1}{m_i}\sum_{j=1}^{m_i} L_{ij}\mathbf{r}_{ij}^T\Big)\mathbf{b} \tag{8}$$

The normalization by m_i takes care of unequal numbers of candidate tag sequences that often arises because of the difference in the lengths of the input sentences. Now let R denote a matrix of the same size as that of Y (*i.e.* $d_2 \times n$) with its i^{th} column as given by $\frac{1}{m_i}\sum_{j=1}^{m_i} L_{ij}\mathbf{r}_{ij}$, then we add the following term to the generative objective function:

$$\sum_{i=1}^{n}\mathbf{a}^T\mathbf{x}_i\Big(\frac{1}{m_i}\sum_{j=1}^{m_i} L_{ij}\mathbf{r}_{ij}^T\Big)\mathbf{b} = \mathbf{a}^T X R^T \mathbf{b} \tag{9}$$

Finally, the projection directions are obtained by solving the following optimization problem :

$$\arg\max_{\mathbf{a},\mathbf{b}} \; (1-\lambda)\mathbf{a}^T X Y^T \mathbf{b} + \lambda\, \mathbf{a}^T X R^T \mathbf{b} \tag{10}$$
$$\text{s.t. } \mathbf{a}^T X X^T \mathbf{a} = 1 \quad \text{and} \quad \mathbf{b}^T Y Y^T \mathbf{b} = 1$$

where $0 \leq \lambda \leq 1$ is the weight parameter to be tuned on the development set.

3 Optimization

In this section, we describe how we solve the optimization problems associated with our models. First we discuss the solution of the generative model. Next, we discuss the *softened* discriminative model, since its solution will be used as a subroutine in our final discussion of the fully discriminative model.

3.1 Optimizing the Generative Model

The optimization problem corresponding to the generative model turns out to be identical to that of canonical correlation analysis (CCA) (Hotelling, 1936; Hardoon et al., 2004), which immediately suggests a solution by solving an eigensystem. In particular, the projection directions are obtained by solving the following generalized eigensystem:

$$\begin{pmatrix} 0 & C_{xy} \\ C_{yx} & 0 \end{pmatrix}\begin{pmatrix}\mathbf{a}\\ \mathbf{b}\end{pmatrix} = \begin{pmatrix} C_{xx} & 0 \\ 0 & C_{yy}\end{pmatrix}\begin{pmatrix}\mathbf{a}\\ \mathbf{b}\end{pmatrix} \tag{11}$$

where $C_{xx} = (1-\tau)XX^T + \tau I$, $C_{yy} = (1-\tau)YY^T + \tau I$ are autocovariance matrices, $C_{xy} = XY^T$ is the cross-covariance matrix, $C_{yx} = C_{xy}^T$, τ is a regularization parameter and I is the identity matrix of appropriate size. Using these eigenvectors as columns, we form projection matrices A and B. These projection matrices are used to project sentences and tag sequences into a common lower dimensional subspace. In general, using all the eigenvectors is sub-optimal from the generalization perspective so we retain only top k eigenvectors.

3.2 Optimizing the Softened Model

In the softened discriminative version, the summation of all the difference terms over all candidate tag sequences and sentences (Eq. 9), enables a simpler objective function whose optimum can be derived by following a procedure very similar to that of the generative model. In particular, the projection directions are obtained by solving Eq. 11 except that C_{xy} is replaced with $X((1-\lambda)Y^T + \lambda R^T)$.

3.3 Optimizing the Discriminative Model

To solve the discriminative model, we begin by constructing the Lagrange dual. Let β_1, β_2 and α_{ij} be the Lagrangian multipliers corresponding to the length and the margin constraints respectively, then the Lagrangian of Eq. 6 is given by:

$$\begin{aligned}\mathcal{L} &= \frac{1-\lambda}{\lambda}\,\mathbf{a}^T X Y^T \mathbf{b} - \sum_{i=1}^{n} \xi_i \\ &\quad - \beta_1\Big(\mathbf{a}^T X X^T \mathbf{a} - 1\Big) - \beta_2\Big(\mathbf{b}^T Y Y^T \mathbf{b} - 1\Big) \\ &\quad + \sum_{i=1,j=1}^{n,m_i} \alpha_{ij}\left(\mathbf{a}^T \mathbf{x}_i \mathbf{r}_{ij}^T \mathbf{b} - 1 + \frac{\xi_i}{L_{ij}}\right)\end{aligned}$$

Differentiating the Lagrangian with respect to the parameters $\mathbf{a}, \mathbf{b}$ and setting them to zero yields the solution for the parameters in terms of the Lagrangian multipliers α_{ij} as follows:

$$\begin{pmatrix} 0 & C^{\alpha}_{xy} \\ C^{\alpha}_{yx} & 0 \end{pmatrix}\begin{pmatrix}\mathbf{a}\\ \mathbf{b}\end{pmatrix} = \begin{pmatrix} C_{xx} & 0 \\ 0 & C_{yy}\end{pmatrix}\begin{pmatrix}\mathbf{a}\\ \mathbf{b}\end{pmatrix} \quad (12)$$

where $C^{\alpha}_{xy} = X\Big(\frac{1-\lambda}{\lambda} Y^T + R^T\Big)$ and R is a matrix of size $d_2 \times n$ with i^{th} column as given by $\frac{1}{m_i}\sum_{j=1}^{m_i} \alpha_{ij}\mathbf{r}_{ij}$. We use superscript α on the cross-covariance matrix to indicate that it is dependent on the Lagrangian multipliers α_{ij}. In other words, the solution is similar to that of the previous formulation except that the residual vectors are weighted by the Lagrangian multipliers instead of the loss function. Unlike the max margin formulations of SVM, it is not easy to rewrite the parameters $\mathbf{a}, \mathbf{b}$ in terms of the Lagrangian multipliers α_{ij} as C^{α}_{xy} itself depends on α_{ij}'s. Hence, rewriting the parameters in terms of the Lagrangian multipliers and then solving the dual is not amenable in this case.

In order to solve this optimization problem, we resort to an alternate optimization technique in the primal space. It proceeds in two stages. In the first stage, we keep the Lagrangian multipliers α_{ij} fixed and then solve for the parameters $\mathbf{a}, \mathbf{b}, \beta_1, \beta_2$ and ξ_i. Projection directions $\mathbf{a}, \mathbf{b}$ and their Lagrangian multipliers β_1, β_2 are obtained by solving the generalized eigenvalue problem given in Eq. 12. Using these projection directions, we determine the slack variable ξ_i for each sentence. In the second stage of the alternate optimization, we fix $\mathbf{a}, \mathbf{b}$ and ξ_i and take a gradient descent step along α_{ij}'s to minimize the function. We repeat this process until convergence. In our experiments, we noticed that this algorithm converges within five iterations, so we only run it for five iterations.

Algorithm 1 Alternate optimization algorithm for solving the parameters of Discriminative Model.

```
Input: $X, Y, \hat{Y}, L, \lambda, \tau$
Output: $A, B$
 1: $\forall i, j \quad \alpha_{ij} = L_{ij}$;
 2: $\mathbf{r}_{ij} = \mathbf{y}_i - \hat{\mathbf{y}}_{ij}$; $C_{xx} = (1-\tau)XX^T + \tau I$;
    $C_{yy} = (1-\tau)YY^T + \tau I$
 3: repeat
 4:   Form $R$ with $i^{th}$ column as $\frac{1}{m_i}\sum_{j=1}^{m_i}\alpha_{ij}\mathbf{r}_{ij}$
 5:   $C^{\alpha}_{xy} = X\Big(\frac{1-\lambda}{\lambda}\, Y^T + R^T\Big)$
 6:   Solve for the eigenvectors of Eq. 12. .
 7:   Form matrices $A, B$ with top $k$ eigenvectors
      as columns; $k$ is determined using dev. set.
 8:   Let $A_n$ & $B_n$ be normalized versions of $A$
      and $B$ s.t. they follow the length constraints.
 9:   for each sentence $i = 1 \cdots n$ do
10:     $j = 1 \cdots m_i$, $\psi_{ij} = \big(1 - \mathbf{x}_i^T A_n B_n^T \mathbf{r}_{ij}\big) L_{ij}$
11:     $\xi_i = \min\big\{0\ ,\ \psi_{ij} \mid \text{s.t. } \psi_{ij} > 0\big\}$
12:     if $\xi_i > 0$ then
13:       $d_{ij} = \mathbf{x}_i^T A_n B_n^T \mathbf{r}_{ij} - \big(1 - \frac{\xi_i}{L_{ij}}\big)$
14:       $\alpha_{ij} = \alpha_{ij} - \gamma\, d_{ij}$
15:     end if
16:   end for
17: until slack values doesn't change
18: return $A, B$
```

The pseudocode of our approach is shown in Alg. 1. First we initialize the Lagrangian multipliers proportional to the loss of the candidate tag sequences (step 1). This ensures that the eigenvectors solved in step 6 are same as the output given by the softened model (Sec. 2.5). In general, in our experiments, we observed that this is a good starting point. After solving the generalized eigenvalue problem in step. 6, we consider the top k eigenvectors, as determined by the error on the development set and normalize them so that they follow the length constraints (steps 7 and 8). In the rest of the algorithm, we use these normalized projection directions to find the slack values which are in turn used to find the update direction for the Lagrangian variables.

In step 10, we compute the potential slack value (ψ_{ij}) for each constraint so that it is satisfied and then choose the minimum of the positive ψ_{ij} values as the slack for this sentence (step 11). If the chosen slack value is equal to zero, it implies that $\psi_{ij} \leq 0 \; \forall j = 1 \cdots m_i$ which in turn implies that all the constraints of a given input sentence are satisfied by the current projection directions and hence there is no need to update the Lagrangian multipliers. Otherwise, some of the constraints are still not satisfied and hence we will update their corresponding Lagrangian multipliers in steps 13 and 14. In specific, step 13 computes the deviation of the margin constraints with the new slack value and step 14 updates the Lagrangian multipliers along the gradient direction.

In principle, our approach is similar to the cutting plane algorithm used to optimize slack re-scaling version of Structured SVM (Tsochantaridis et al., 2004), but it differs in selecting the slack variable (step 11). The cutting plane method chooses ξ_i as the maximum of $\{0, \psi_{ij}\}$ where as we choose the minimum of the positive ψ_{ij} values as the slack. Intuitively, this means that the cutting plane algorithm chooses a constraint that is most violated which results in fewer constraints. This is crucial in structured SVM, because solving the dual problem is cubic in terms of the number of examples and constraints. In contrast, our approach selects the slack such that at least one of the constraints is satisfied and adds all the remaining constraints to the active set. Since step 6 considers a weighted average of all these constraints the complexity depends only on the number of training examples and not the constraints.

3.4 Combining with Viterbi Decoding Score

All the three formulations discussed until now do not consider the Viterbi decoding score assigned to each candidate tag sequence. As explained in Collins and Koo (2005), the decoding score plays an important role in reranking the candidate sentences. Here, we describe a simple linear combination of the Viterbi decoding score and the score obtained by projecting into the low-dimensional subspace, using projection directions obtained by any of the above models.

For a given sentence $\mathbf{x}_i$ and candidate tag sequence pair $\hat{\mathbf{y}}_{ij}$, let s_{ij} and p_{ij} (Eq. 1) be the scores assigned by Viterbi decoding and the lower dimensional projections respectively. Then we define the final score for this pair as a simple linear combination of these two scores as:

$$\text{Score}(\mathbf{x}_i, \hat{\mathbf{y}}_{ij}) = s_{ij} + w\, p_{ij} \qquad (13)$$

The weight w is optimized using a grid search on the development data set, we search for w from 0 to 100 with an increment of 1 and choose the value for which the error is minimum on the development set.

3.5 Reranking for POS Tagging

To summarize our approach, we convert the training data into feature vectors and use any of the three methods discussed above to find the lower dimensional projection directions ($\mathbf{a}$ and $\mathbf{b}$). Each of those approaches involve solving a similar generalized eigenvalue problem (Eq. 11) with the cross covariance matrix C_{xy} defined differently in the three approaches. This problem can be solved in different ways, but we use the following approach since it reduces the size of the eigenvalue problem.

$$C_{yy}^{-1} C_{xy}^{T} C_{xx}^{-1} C_{xy}\, \mathbf{b} = \omega\, \mathbf{b} \qquad (14)$$

$$\mathbf{a} = \frac{1}{\sqrt{\omega}}\, C_{xx}^{-1} C_{xy}\, \mathbf{b} \qquad (15)$$

where ω is the eigenvalue. Assuming that $d_2 \ll d_1$, which is usually true in POS tagging because of the smaller tag vocabulary, these equations solve a smaller eigenvalue problem. After solving the eigenvalue problem, we form matrices A and B with columns as the top k eigenvectors $\mathbf{a}$ and $\mathbf{b}$ respectively. Given a new sentence and candidate tag sequence pair $(\mathbf{x}_i, \hat{\mathbf{y}}_{ij})$, their similarity is obtained using Eq. 1. Now, based on the development data set we find the weight (w) for the linear combination of the projection and Viterbi decoding scores (Eq. 13).

During the reranking stage, we first use Eq. 1 to compute the projection score for all the candidate tag sequences and then use Eq. 13 to combine this scores with the decoding score. The candidate tag sequences are reranked based on this final score.

4 Experiments

In this section, we report POS tagging experiments on four languages: English, Chinese, French and

		Train.	Dev.	Test
English (En.)	# sent.	15K	2K	1791
	# words	362K	47K	43K
Chinese (Zh.)	# sent.	50K	4K	3647
	# words	292K	26K	25K
French (Fr.)	# sent.	9K	2K	1351
	# words	254K	57K	40K
Swedish (Sv.)	# sent.	8K	2K	1431
	# words	137K	31K	28K

Table 1: Training and test data statistics.

Swedish. The data in all these languages is obtained from the CoNLL 2006 shared task on multilingual dependency parsing (Buchholz and Marsi, 2006). We only consider the word and its fine grained POS tag (columns 2 and 5 respectively) and ignore the dependency links in the data. Table 1 shows the data statistics in each of these languages.

We use a second order Hidden Markov Model (Thede and Harper, 1999) based tagger as a baseline tagger in our experiments. This model uses trigram transition and emission probabilities and is shown to achieve good accuracies in English and other languages (Huang et al., 2007). We refer to this as the baseline tagger in the rest of this paper and is used to produce n-best list for each candidate sentence. The n-best list for training data is produced using multi-fold cross-validation like Collins and Koo (2005) and Charniak and Johnson (2005). The first block of Table 2 shows the accuracies of the top-ranked tag sequence (according to the Viterbi decoding score) and the oracle accuracies on the 10-best list. As expected the accuracies on English and French are high and are on par with the state-of-the-art systems. From the oracle scores, it is clear that though there is a chance for improvement using reranking, the scope for improvement in English is less compared to the 5 point improvement reported for parsing (Charniak and Johnson, 2005). This indicates the difficulty of the reranking problem for POS tagging in well-resourced languages.

4.1 Reranking Features and Baselines

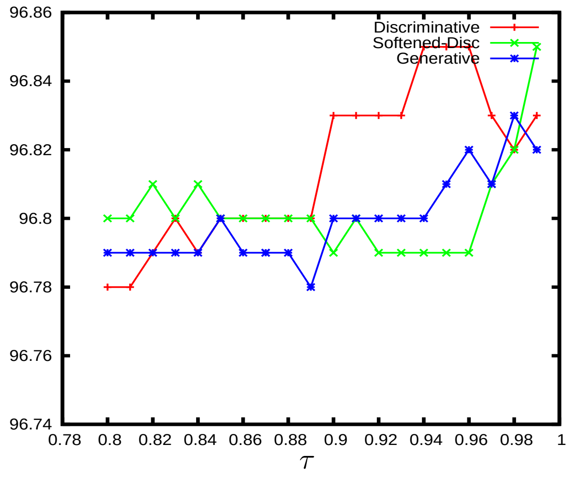
In this paper, except for Chinese, we use suffixes of length two to four as features in the word view and unigram and bigram tag sequences as features in the tag view. That is, we convert each word of the sentence into suffixes of length two to four and then treat each sentence as a bag of suffixes. Similarly, we treat a candidate POS tag sequence as a bag of unigram and bigram tag features. For Chinese, we use character sequences of length one and two as features for the sentences and use unigram and bigram POS tag sequences on the tag view. We did not include any alignment based features, *i.e.* features that depend on the position.

We compare our models with a boosting-based discriminative approach (Collins and Koo, 2005) and its regularized version (Huang et al., 2007). In order to enable a fair comparison, we use suffix and tag pairs as features for both these models. For example, we would generate the following features for the word ‘selling’ in the phrase “the/DT selling/NN pressure/NN”: (ng, NN), (ng, DT_NN), (ing,NN), (ing,DT_NN), (ling,NN), (ling,DT_NN). For comparison purposes, we also show results by running the baseline rerankers with n-gram features.

4.2 Results

There are following hyper parameters in each of our models, regularization parameter τ, weight parameter λ in the discriminative and softened discriminative models, the linear combination weight w with the Viterbi decoding score, and finally, the size of the lower dimensional subspace (k). We use grid search to tune these parameters based on the development data set. The optimal hyperparameter values differ based on the model and the language, but the tagging accuracy is relatively robust with respect to these parameter values. For English, the best values for the discriminative model are $\tau = 0.95$, $\lambda = 0.3$ and $k = 75$. For the same language, Fig. 1 shows the performance with respect to τ and λ parameters, respectively, with other parameters fixed to their optimal values. Notice that, although the performance varies it is always more than the accuracy of the baseline tagger (96.74%).

Table 2 shows the results of different models on the development and test data sets. On the test data set, the baseline reranking approaches perform better than the HMM decoder in Chinese and Swedish languages, but they underperform in English and French languages. This is justifiable because the individual characters are good indicators of POS tag



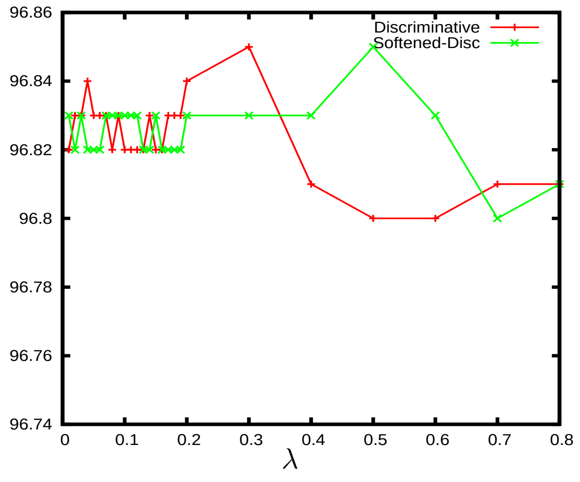


Figure 1: Tagging accuracy with hyperparameters τ and λ on English development data set.

	Development Set				Test set			
	English	Chinese	French	Swedish	English	Chinese	French	Swedish
Baseline	96.74	92.55	96.94	93.22	96.15	92.31	97.41	93.23
Oracle	98.85	98.41	98.61	96.96	98.39	98.19	99.00	96.48
Collins (Sufx)	96.66	93.00	96.87	**93.50**	96.06	92.81	97.35	**93.44**
Regularized (Sufx)	96.60	93.12	96.90	93.36	96.00	92.88	97.38	93.35
Generative	96.82	93.14	96.97	93.46	96.24	**92.95**	97.43	93.26
Softened-Disc	96.85	93.14	**97.04**	93.49	**96.32**	92.87	**97.53**	93.24
Discriminative	**96.85**	**93.17**	97.03	**93.50**	96.3	92.91	**97.53**	93.36
Collins (n-gm)	96.74	93.14	97.06	93.44	96.13	92.74	97.54	93.45
Regularized (n-gm)	96.78	93.14	97.01	93.45	96.14	92.80	97.52	93.40

Table 2: Accuracy of the baseline HMM tagger and different reranking approaches. For comparison purposes, we also showed the results of Collins and Koo (2005) its regularized versions with n-gram features. The improvements of our discriminative models are statistically significant at $p = 0.01$ and $p = 0.05$ levels on Chinese and English respectively.

information for Chinese and this additional information is being exploited by the reranking approaches. Swedish, on the other hand, is a Germanic language with compound word phenomenon which makes the baseline HMM decoder weaker compared to English and French.

The fourth block shows the performance of our models. Except in Swedish, one of our models outperform the baseline decoder and the other reranking approaches. The fact that our models outperform the baseline system and other reranking approaches indicate that, by considering all the pairwise combinations of the input features our models capture dependencies that are left by other models. Among the different formulations of our approach, maximizing the margin between the correct and incorrect candidates performed better than generative, and ensuring that the margin is proportional to the loss of the candidate sequence (discriminative) led to even more improved results. Except in Chinese, our discriminative version performed at least as well as the other variants. Compared to the baseline decoder, the discriminative version achieves a maximum improvement of 0.6 points in Chinese while achieving 0.15, 0.12 and 0.13 points of improvement in English, French and Swedish languages respectively.

We also reported the results of the baseline rerankers with n-gram features in the fifth block of Table 2. We remind the reader that our models use only suffix features, so for a fair comparison the

	En.	Zh.	Fr.	Sv.
Generative	94.83	89.89	96.1	91.89
Softened-Disc	95.04	89.61	95.97	91.95
Discriminative	94.95	89.76	95.82	92.11

Table 3: Accuracies without combining with Viterbi decoding score.

reader should compare our results with the baseline rerankers run with the suffix features. The performance of these baseline rankers improved when we include the n-gram features but it is still less than the discriminative model in most cases.

Finally, Table 3 shows the performance of our models without combining with the Viterbi decoding score. As shown, the performance drops significantly and is in accordance with the behavior observed elsewhere (Collins and Koo, 2005).

5 Related Work

In this section, we discuss approaches that are most relevant to our problem and the approach.

In NLP literature, discriminative reranking has been well explored for parsing (Collins and Koo, 2005; Charniak and Johnson, 2005; Shen and Joshi, 2003; McDonald et al., 2005; Johnson and Ural, 2010) and statistical machine translation (Shen et al., 2004; Watanabe et al., 2007; Liang et al., 2006). Collins (2002) proposed two reranking approaches, namely boosting algorithm and a voted perceptron, for the POS tagging task. Later Huang *et al.* (2007) propose a regularized version of the objective used by Collins (2002) and show an improved performance for Chinese. In all of the above reranking approaches, the feature functions are defined jointly on the input and output, whereas in our approach, the features are defined separately within each view and the algorithm learns the relationship between them automatically. This is the primary difference between our approach and the existing rerankers.

In principle, our margin formulations are similar to the max margin formulations of CCA (Szedmak et al., 2007) and maximum margin regression (Szedmak et al., 2006; Wang et al., 2007). These approaches solve the following optimization problem:

$$\begin{aligned} \min \ & \|W\|^2 + C\mathbf{1}^T\boldsymbol{\xi} \qquad (16)\\ \text{s.t.} \ & \langle \mathbf{y}_i, W\phi(\mathbf{x})_i\rangle \geq 1 - \xi_i \quad \forall i = 1\cdots n \end{aligned}$$

Our approach differs from these formulations in two main ways: the score assigned by our generative model (equivalent to CCA) for an input-output pair ($\mathbf{x}_i^T\mathbf{a}\mathbf{b}^T\mathbf{y}_i$) can be converted into this format by substituting $W \leftarrow \mathbf{b}\mathbf{a}^T$ but in doing so we are ignoring the rank constraint. It is often observed that, dimensionality reduction leads to an improved performance and thus the rank constraint becomes crucial. Another major difference is that, the constraints in Eq. 16 represent that any input and output pair should have at least a margin of 1 (modulo slack), whereas in our approach, the constraints include incorrect outputs along with their loss value. In other words, our formulation is more suitable for the reranking problem while Eq. 16 is more suitable for regression or classification tasks. Our generative model is very similar to the supervised semantic hashing work (Bai et al., 2010) but the way we optimize is completely different from theirs.

6 Discussion

In this paper, we proposed a novel family of models for discriminative reranking problem and showed improvements for the POS tagging task in four different languages. Here, we restricted our scope to showing the utility of our technique and, hence, did not experiment with different features, though it is an important direction. By using only within space features, our models are able to beat the reranking approaches that use potentially more informative alignment-based features. It is also possible to include alignment-based features into our models by posing the problem as a feature selection problem on the covariance matrices (Jagarlamudi et al., 2011). Our approach involves an inverse computation and an eigenvalue problem. Although our models scale to medium size data sets (our Chinese data set has 50K examples and 33K features), these operations can be expensive. But there are alternative approximation techniques that scale well to large data sets (Halko et al., 2009). We leave this for future work.

Acknowledgments

We thank Zhongqiang Huang for providing the code for the baseline systems, Raghavendra Udupa and the anonymous reviewers for their insightful comments. This work is partially funded by NSF grants IIS-1153487 and IIS-1139909.

References

Bing Bai, Jason Weston, David Grangier, Ronan Collobert, Kunihiko Sadamasa, Yanjun Qi, Olivier Chapelle, and Kilian Weinberger. 2010. Learning to rank with (a lot of) word features. *Inf. Retr.*, 13(3):291–314, June.

Gükhan H. Bakir, Thomas Hofmann, Bernhard Schölkopf, Alexander J. Smola, Ben Taskar, and S. V. N. Vishwanathan. 2007. *Predicting Structured Data (Neural Information Processing)*. The MIT Press.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Comput. Linguist.*, 19:263–311, June.

Sabine Buchholz and Erwin Marsi. 2006. Conll-x shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, CoNLL-X '06, pages 149–164, Stroudsburg, PA, USA. Association for Computational Linguistics.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 173–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

David Chiang, Kevin Knight, and Wei Wang. 2009. 11,001 new features for statistical machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 218–226, Stroudsburg, PA, USA. Association for Computational Linguistics.

Michael Collins and Terry Koo. 2005. Discriminative reranking for natural language parsing. *Computational Linguistics*, 31:25–70, March.

Michael Collins. 2002. Ranking algorithms for named-entity extraction: boosting and the voted perceptron. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 489–496, Stroudsburg, PA, USA. Association for Computational Linguistics.

Alexander Fraser, Renjing Wang, and Hinrich Schütze. 2009. Rich bitext projection features for parse reranking. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '09, pages 282–290, Stroudsburg, PA, USA. Association for Computational Linguistics.

Nathan Halko, Per-Gunnar. Martinsson, and A. Joel Tropp. 2009. Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions. Technical report, California Institute of Technology.

David R. Hardoon, Sandor R. Szedmak, and John R. Shawe-taylor. 2004. Canonical correlation analysis: An overview with application to learning methods. *Neural Comput.*, 16:2639–2664, December.

Harold Hotelling. 1936. Relation between two sets of variables. *Biometrica*, 28:322–377.

Zhongqiang Huang, Mary Harper, and Wen Wang. 2007. Mandarin part-of-speech tagging and discriminative reranking. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 1093–1102, Prague, Czech Republic, June. Association for Computational Linguistics.

Jagadeesh Jagarlamudi, Raghavendra Udupa, Hal Daumé III, and Abhijit Bhole. 2011. Improving bilingual projections via sparse covariance matrices. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 930–940, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Mark Johnson and Ahmet Engin Ural. 2010. Reranking the Berkeley and Brown parsers. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 665–668, Stroudsburg, PA, USA. Association for Computational Linguistics.

Sandra Kubler, Ryan McDonald, Joakim Nivre, and Graeme Hirst. 2009. *Dependency Parsing*. Morgan and Claypool Publishers.

Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar. 2006. An end-to-end discriminative approach to machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 761–768, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 91–98, Stroudsburg, PA, USA. Association for Computational Linguistics.

Libin Shen and Aravind K. Joshi. 2003. An SVM based voting algorithm with application to parse reranking. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 9–16, Stroudsburg, PA, USA. Association for Computational Linguistics.

Libin Shen, Anoop Sarkar, and Franz Och. 2004. Discriminative reranking for machine translation. In *Human Language Technology Conference and the 5th Meeting of the North American Association for Computational Linguistics: HLT-NAACL 2004*, Boston, USA, May.

S. Szedmak, J. Shawe-Taylor, and E. Parado-Hernandez. 2006. Learning via linear operators: Maximum margin regression; multiclass and multiview learning at one-class complexity. Technical report, University of Southampton.

Sandor Szedmak, Tijl De Bie, and David R. Hardoon. 2007. A metamorphosis of canonical correlation analysis into multivariate maximum margin learning. In *Proceedings of the fifteenth European Symposium on Artificial Neural Networks*.

Ben Taskar, Carlos. Guestrin, and Daphne Koller. 2004. Max margin markov networks. In *Proceedings of NIPS 16*.

Scott M. Thede and Mary P. Harper. 1999. A second-order Hidden Markov Model for part-of-speech tagging. In *Proceedings of the Annual Meeting on Association for Computational Linguistics*, pages 175–182. Association for Computational Linguistics.

Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, and Yasemin Altun. 2004. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the twenty-first international conference on Machine learning*, ICML '04, pages 104–, New York, NY, USA. ACM.

Zhuoran Wang, John Shawe-Taylor, and Sandor Szedmak. 2007. Kernel regression based machine translation. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, NAACL-Short '07, pages 185–188, Stroudsburg, PA, USA. Association for Computational Linguistics.

Taro Watanabe, Jun Suzuki, Hajime Tsukada, and Hideki Isozaki. 2007. Online Large-Margin Training for Statistical Machine Translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 764–773, Prague, Czech Republic, June. Association for Computational Linguistics.