A Separately Passive-Aggressive Training Algorithm for Joint POS Tagging and Dependency Parsing

Zhenghua Li[1], Min Zhang[2], Wanxiang Che[1], Ting Liu[1]*
(1) Research Center for Social Computing and Information Retrieval,
Harbin Institute of Technology, China
(2) Institute for Infocomm Research, Singapore
{lzh,car,tliu}@ir.hit.edu.cn, mzhang@i2r.a-star.edu.sg

ABSTRACT
Recent study shows that parsing accuracy can be largely improved by the joint optimization of part-of-speech (POS) tagging and dependency parsing. However, the POS tagging task does not benefit much from the joint framework. We argue that the fundamental reason behind is because the POS features are overwhelmed by the syntactic features during the joint optimization, and the joint models only prefer such POS tags that are favourable solely from the parsing viewpoint. To solve this issue, we propose a separately passive-aggressive learning algorithm (SPA), which is designed to separately update the POS features weights and the syntactic feature weights under the joint optimization framework. The proposed SPA is able to take advantage of previous joint optimization strategies to significantly improve the parsing accuracy, but also overcome their shortages to significantly boost the tagging accuracy by effectively solving the syntax-insensitive POS ambiguity issues. Experiments on the Chinese Penn Treebank 5.1 (CTB5) and the English Penn Treebank (PTB) demonstrate the effectiveness of our proposed methodology and empirically verify our observations as discussed above. We achieve the best tagging and parsing accuracies on both datasets, 94.60% in tagging accuracy and 81.67% in parsing accuracy on CTB5, and 97.62% and 93.52% on PTB.

KEYWORDS: Part-of-speech Tagging, Dependency Parsing, Joint Models, Separately Passive-aggressive Algorithm.

* Corresponding author

Proceedings of COLING 2012: Technical Papers, pages 1681–1698,
COLING 2012, Mumbai, December 2012.

1 Introduction

Given an input sentence of n words, denoted by $\mathbf{x} = w_1...w_n$, part-of-speech (POS) tagging aims to find an optimal tag sequence $\mathbf{t} = t_1...t_n$, where $t_i \in \mathcal{T}$ $(1 \leq i \leq n)$ and $\mathcal{T}$ is a predefined tag set. POS tags are designed to represent word classes so that words of the same POS tag play a similar role in syntactic structures. The size of $\mathcal{T}$ is usually much less than the vocabulary size. Typically, POS tagging is treated as a sequence labeling problem, and has been previously addressed by machine learning algorithms, such as maximum-entropy (Ratnaparkhi, 1996), conditional random fields (CRF) (Lafferty et al., 2001) and perceptron (Collins, 2002). Figure 1 gives an example sentence from Penn Chinese Treebank 5.1 (CTB5). The lowest three rows present the n-best POS tags for each word, produced by a state-of-the-art CRF model. Looking at the 1-best POS tags, we can see that the CRF model makes four errors, i.e. de/DEC→DEG, ouwen/NR→NN, xiaoli/VV→NN, and liwupudui/NR→NN. In fact, (DEC,DEG) and (NN,VV) ambiguities, which usually require long-distance syntactic knowledge to resolve, are very difficult for the sequential labeling models.

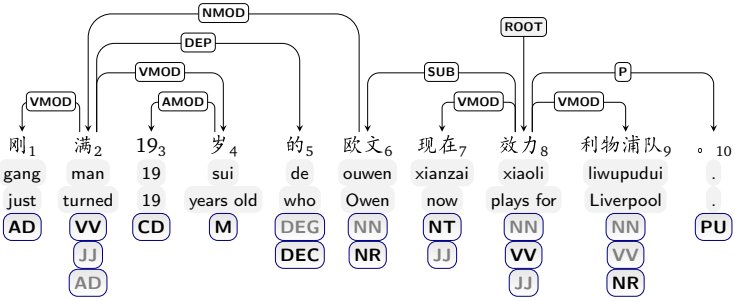


Figure 1: An example from CTB5. The Pinyin transcriptions and English translations are presented in the two rows below the Chinese sentence. We prune out the implausible POS tags according to the marginal probabilities (see Section 4.1) and list the top three candidate POS tags in the lowest three rows (incorrect POS tags in grey color and correct ones in black color).

Dependency Parsing maps a natural language sentence into a structural dependency tree conforming to a predefined dependency grammar, as depicted in Figure 1. A dependency tree is denoted by $\mathbf{d} = \{(h, m, l) : 1 \leq h \leq n, 1 \leq m \leq n, l \in \mathcal{L}\}$, where (h, m, l) means a dependency from the *head word* (also called *father*) w_h to the *modifier* (also called *child* or *dependent*) w_m with a dependency label l, and $\mathcal{L}$ is the label set. Dependency labels are used to indicate the syntactic or semantic relation between the two words. For instance, the dependency (8, 6, SUB) in Figure 1 means ouwen is the *subject* of xiaoli.

Data-driven dependency parsing models make heavy use of POS tags to compose supporting features, since it leads to severe data sparseness problem if only lexical features are used. However, POS tagging errors significantly degrade the parsing accuracy by about 6% (see Table 4 where, for example, due to the error of xiaoli/VV→NN, our pipelined parsing model fails to recognize xiaoli as the predicate of the sentence and returns a fully unreasonable structure).

Recently, there has been increasing interest in joint modeling of Chinese POS tagging and

dependency parsing (Li et al., 2011; Hatori et al., 2011; Bohnet and Nivre, 2012), motivated by the intuition that the two individual tasks should help each other. Their work demonstrates that the joint models can substantially boost the parsing accuracy. In contrast, the tagging subtask does not benefit much from the joint framework. (Li et al., 2011) show that the graph-based joint models lead to large decrease in the tagging accuracy, whereas (Hatori et al., 2011) and (Bohnet and Nivre, 2012) find small gains in the tagging accuracy with transition-based joint models (see Table 4). This is contradictory to the intuition that better syntactic structure should help POS disambiguation. The detailed error analysis in (Li et al., 2011) show that their joint models are helpful in resolving syntax-sensitive POS ambiguities like {VV,NN} and {DEC,DEG}, but become very weak in disambiguating {NN,NR} and {NN,JJ} which usually play similar roles in syntactic trees.

We believe that one possible reason is that the graph-based joint models of (Li et al., 2011) is dominated by the syntactic features. Looking deeper into their joint models, we find that on average, the score corresponding to the POS features only is $1/50$ of the score of the syntactic features in a returned joint result. In other words, the POS features have little impact on determining the best joint result. Therefore, the joint models prefer such POS tags that are more helpful and discriminative solely from the parsing viewpoint.

To address this issue, this paper proposes a variant of the passive-aggressive (PA) online training algorithm (Crammer et al., 2003), which we name as *separately passive-aggressive algorithm (SPA)*. SPA separately updates the POS feature weights and the syntactic feature weights and naturally raises the weights of the POS features under the joint optimization framework. As a result, SPA can make better use of the discriminative power of the POS features in resolving the syntax-insensitive POS ambiguities, leading to a large tagging accuracy improvement. On the other hand, the improved tagging accuracy can further help parsing. Specifically, we make the following contributions.

- We propose a separately passive-aggressive training algorithm for joint POS tagging and dependency parsing. Empirically, we compare SPA with averaged perceptron (AP) and PA from the perspective of the model score, showing that SPA is more suitable for the joint models. Experimental results demonstrate that SPA outperforms AP and PA in both the tagging and parsing accuracies. More importantly, SPA significantly improve both the tagging and parsing accuracies over the pipelined baselines.
- We present the first feature-rich graph-based joint model for POS tagging and labeled dependency parsing. We conduct experiments on two versions of CTB5 and achieve the best tagging and parsing accuracies on both datasets. Especially, our joint model trained with SPA achieves a tagging accuracy of 94.7%, largely outperforming the 93.9% of the baseline CRF model.
- We also conduct experiments on PTB to find out the effect of joint modeling on English. Thanks to relatively richer morphologies, English POS tagging achieves much higher accuracy than Chinese (97% vs. 94%). Therefore, English dependency parsing suffers less error propagation problem than Chinese. Nevertheless, we still find significant gains from the joint model. The tagging accuracy and the parsing accuracy increase by about 0.5% and 0.4% respectively. Our joint model achieves the best tagging and parsing accuracies on this dataset as well.
- We present the first work that conducts extensive studies on the effect of labeled dependency parsing. We find that dependency labels consistently improve the unlabeled parsing accuracy. Especially, the unlabeled attachment score (UAS) can be boosted by

0.6-0.7% on the two Chinese datasets.

2 Related work

This work is most closely related to (Li et al., 2011) who present the first work on joint models for Chinese POS tagging and unlabeled dependency parsing. Similar to us, their joint models are based on graph-based dependency parsing. They find that the joint models largely outperform the pipeline models in the parsing accuracy but lead to substantial tagging accuracy drop. Compared with their work, we propose a better training algorithm for the joint models that can improve both tagging and parsing accuracies. In addition, our joint model adopts richer features and handles labeled dependency parsing.

(Hatori et al., 2011) propose the first transition-based joint model for Chinese POS tagging and unlabeled dependency parsing and gain large improvement in the parsing accuracy. However, their joint models only slightly improve the tagging accuracy over a sequential tagging model. (Bohnet and Nivre, 2012) propose a transition-based joint model which can handle labeled non-projective dependency parsing. They conduct experiments on a variety of languages including Chinese, English, Czech, and German. Similarly, their joint model largely improves the parsing accuracy but only slightly increases the tagging accuracy. Differently, we are the first work on joint POS tagging and dependency parsing that achieves large improvement in the tagging accuracy.

(Smith and Eisner, 2008) apply loopy belief propagation (LBP) to dependency parsing and points out that LBP can naturally represent POS tags as latent variables so that the POS tags can be inferred jointly with the parse. (Lee et al., 2011) extend the LBP based approach of (Smith and Eisner, 2008) and study joint morphological disambiguation and dependency parsing for morphologically-rich languages including Latin, Czech, Ancient Greek, and Hungarian. For these languages, morphological analysis requires the disambiguation of POS tags, gender, case, etc. They show that the joint model can well capture the interaction between morphology and syntax and achieve gains on both subtasks. (Rush et al., 2010) propose dual decomposition (DD) for integrating different NLP subtasks at the test phase. They experiment with two cases, one integrating a phrase-structure parser and a dependency parser, and the other integrating a phrase-structure parser and a POS tagger. Both cases show that DD can help the individual subtasks. (Auli and Lopez, 2011) conduct an extensive comparison of LBP and DD for joint CCG supertagging and parsing. They show that LBP and DD achieves similar parsing accuracy improvement but has largely different convergence characteristics. Moreover, their work focuses on integrating two separately-trained sub-models, and they find that training the integrated model on LBP leads to large improvement drops compared with separately-trained models.

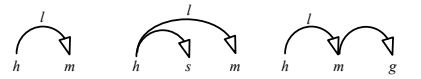
3 Pipeline POS tagging and dependency parsing

The pipeline method treats POS tagging and dependency parsing as two cascaded problems. First, an optimal POS tag sequence $\hat{\mathbf{t}}$ is determined.

$$\hat{\mathbf{t}} = \arg\max_{\mathbf{t}} \mathrm{Score}_{\mathrm{pos}}(\mathbf{x}, \mathbf{t}) \tag{1}$$

Then, an optimal dependency tree $\hat{\mathbf{d}}$ is determined based on $\mathbf{x}$ and $\hat{\mathbf{t}}$.

$$\hat{\mathbf{d}} = \arg\max_{\mathbf{d}} \mathrm{Score}_{\mathrm{syn}}(\mathbf{x}, \hat{\mathbf{t}}, \mathbf{d}) \tag{2}$$



(a) *single dependency* (b) *adjacent sibling* (c) *outermost grandchild*

Figure 2: Three types of scoring subtrees used in our parsing and joint models.

Feature category	Atomic features incorporated
Dependency features $\mathbf{f}_{\text{dep}}(\mathbf{x},\mathbf{t},h,m,l)$	$l, w_h, w_m, t_h, t_m, t_{h\pm1}, t_{m\pm1}, t_b, dir(h,m), dist(h,m)$
Sibling features $\mathbf{f}_{\text{sib}}(\mathbf{x},\mathbf{t},h,m,l,s)$	$l, w_h, w_s, w_m, t_h, t_m, t_s, t_{h\pm1}, t_{m\pm1}, t_{s\pm1}, dir(h,m), dist(h,m)$
Grandchild features $\mathbf{f}_{\text{grd}}(\mathbf{x},\mathbf{t},h,m,l,g)$	$l, w_h, w_m, w_g, t_h, t_m, t_g, t_{h\pm1}, t_{m\pm1}, t_{g\pm1}, dir(h,m), dir(m,g)$

Table 1: Brief illustration of the syntactic features. b is an index between h and m. $dir(i,j)$ and $dist(i,j)$ denote the direction and distance of the dependency (i,j). Please refer to Table 4 of (Bohnet, 2010) for the complete feature list.

CRF-based POS tagging. We adopt the first-order CRF to build our baseline POS tagger. As a conditional log-linear probabilistic model, CRF defines the probability of a tag sequence as

$$\begin{aligned} P(\mathbf{t}|\mathbf{x}) &= \exp(\texttt{Score}_{\text{pos}}(\mathbf{x},\mathbf{t}))/\sum_{\mathbf{t}'}\exp(\texttt{Score}_{\text{pos}}(\mathbf{x},\mathbf{t}')) \\ \texttt{Score}_{\text{pos}}(\mathbf{x},\mathbf{t}) &= \mathbf{w}_{\text{pos}}\cdot\mathbf{f}_{\text{pos}}(\mathbf{x},\mathbf{t}) = \sum_{1\le i\le n}\mathbf{w}_{\text{pu}}\cdot\mathbf{f}_{\text{pu}}(\mathbf{x},t_i) + \mathbf{w}_{\text{pb}}\cdot\mathbf{f}_{\text{pb}}(\mathbf{x},t_{i-1},t_i) \end{aligned} \tag{3}$$

where $\mathbf{f}_{\text{pos/pu/pb}}(.)$ refers to the feature vectors and $\mathbf{w}_{\text{pos/pu/pb}}$ is the corresponding weight vectors. We call $\mathbf{f}_{\text{pu}}(\mathbf{x},t_i)$ the *POS unigram features*, and $\mathbf{f}_{\text{pb}}(\mathbf{x},t_{i-1},t_i)$ the *POS bigram features*. For Chinese, we adopt the features proposed by (Zhang and Clark, 2008a). They use Chinese characters contained in a word to compose rich features, which turns out to be helpful for low-frequency words. For English, we adopt the features of (Ratnaparkhi, 1996) which exploit suffixes and prefixes to improve tagging performance over rare words.

Second-order graph-based dependency parsing. The graph-based approach views dependency parsing as finding a highest scoring tree in a directed graph (McDonald et al., 2005; Carreras, 2007; Koo and Collins, 2010). We adopt the second-order model of (Carreras, 2007) since previous studies show that it leads to best parsing accuracy on a variety of languages (Koo and Collins, 2010; Bohnet, 2010). The score of a dependency tree is factored into scores of the three kinds of subtrees in Figure 2.

$$\begin{aligned} \texttt{Score}_{\text{syn}}(\mathbf{x},\mathbf{t},\mathbf{d}) &= \mathbf{w}_{\text{syn}}\cdot\mathbf{f}_{\text{syn}}(\mathbf{x},\mathbf{t},\mathbf{d}) \\ &= \sum_{\{(h,m,l)\}\subseteq\mathbf{d}}\mathbf{w}_{\text{dep}}\cdot\mathbf{f}_{\text{dep}}(\mathbf{x},\mathbf{t},h,m,l) \\ &+ \sum_{\{(h,m,l),(h,s)\}\subseteq\mathbf{d}}\mathbf{w}_{\text{sib}}\cdot\mathbf{f}_{\text{sib}}(\mathbf{x},\mathbf{t},h,m,l,s) \\ &+ \sum_{\{(h,m,l),(m,g)\}\subseteq\mathbf{d}}\mathbf{w}_{\text{grd}}\cdot\mathbf{f}_{\text{grd}}(\mathbf{x},\mathbf{t},h,m,l,g) \end{aligned} \tag{4}$$

For syntactic features, we adopt those of (Bohnet, 2010) which include three categories corresponding to the three types of scoring subtrees. We summarize the atomic features used in

each feature category in Table 1. For unlabeled parsing and joint models, the label l is omitted. Compared with the syntactic features used in (Li et al., 2011), this feature set explores more context POS tags including $t_{s\pm1}, t_{g\pm1}$. In addition, for Chinese, we use the last character of each word as its lemma, and duplicate each word-related feature by replacing words with lemmas (Che et al., 2012). Experiments on CTB5 show that these lemma-related features can improve our baseline parsing models by 0.3-0.4% in UAS. For English, we use coarse-grained POS tags to duplicate all the feature that depend on POS tags (Koo and Collins, 2010), resulting in a 0.4% UAS gain on PTB.

4 Joint POS tagging and dependency parsing

In the joint framework, we aim to simultaneously solve the two problems.

$$(\hat{\mathbf{t}}, \hat{\mathbf{d}}) = \arg\max_{\mathbf{t},\mathbf{d}} \text{Score}_{\text{joint}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) \tag{5}$$

The score of a tagged dependency tree is the combination of the POS score and the syntactic score that are previously defined in the pipeline models.

$$\begin{aligned} \text{Score}_{\text{joint}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) &= \text{Score}_{\text{pos}}(\mathbf{x}, \mathbf{t}) + \text{Score}_{\text{syn}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) \\ &= \mathbf{w}_{\text{pos}} \cdot \mathbf{f}_{\text{pos}}(\mathbf{x}, \mathbf{t}) + \mathbf{w}_{\text{syn}} \cdot \mathbf{f}_{\text{syn}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) \\ &= \mathbf{w}_{\text{pos}\oplus\text{syn}} \cdot \mathbf{f}_{\text{pos}\oplus\text{syn}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) = \mathbf{w}_{\text{joint}} \cdot \mathbf{f}_{\text{joint}}(\mathbf{x}, \mathbf{t}, \mathbf{d}) \end{aligned} \tag{6}$$

where $\oplus$ denotes vector concatenation. Note that our joint model incorporates the same POS and syntactic features with the pipeline models. Under the joint model, the weights of POS and syntactic features, denoted by $\mathbf{w}_{\text{pos}\oplus\text{syn}}$ or $\mathbf{w}_{\text{joint}}$, are simultaneously learned. Therefore, they can interact with each other to determine an optimal joint result.

4.1 Decoding

Similar to (Li et al., 2011), we extend the parsing algorithm of (Carreras, 2007) using the idea of (Eisner, 2000) and propose a dynamic programming (DP) based decoding algorithm for our joint model. Figure 3 illustrates the basic DP structures and operations. The key idea is to augment the basic DP structures in the parsing algorithm (namely *spans*) with a few POS tags. A span means a partially built structure spanning a sub-sentence. For example, the left-side span in Figure 3(a), which is called an *incomplete span* and is denoted by $I_{(h,m,l)(t_h,t_m)}$, represents a partial tree spanning $w_h...w_m$ with w_h being tagged as t_h and w_m as t_m. The left-side span in Figure 3(b) is a complete span and is denoted by $C^{(e)(t_e)}_{(h,m)(t_h,t_m)}$.

The decoding algorithm works in a bottom-up fashion and combines two smaller spans into a larger one at each step. During combination, the newly-introduced features are incorporated and the score of the span is computed accordingly. For example, the operation in Figure 3(a) introduces five feature sets, i.e. $\mathbf{f}_{\text{dep}}(\mathbf{x}, t_h, t_m, h, m, l)$, $\mathbf{f}_{\text{sib}}(\mathbf{x}, t_h, t_m, t_s, h, m, l, s)$, $\mathbf{f}_{\text{grd}}(\mathbf{x}, t_h, t_m, t_g, h, m, l, g)$, $\mathbf{f}_{\text{pu}}(\mathbf{x}, t_m)$, and $\mathbf{f}_{\text{pb}}(\mathbf{x}, t_r, t_{r+1})$. And the operation in Figure 3(b) introduces one feature set $\mathbf{f}_{\text{grd}}(\mathbf{x}, t_h, t_m, t_g, h, m, l, g)$.[1] Note that in the above syntactic feature functions, several context POS tags are not encoded in the DP structures and therefore are not provided in the parameter lists, including $t_{h\pm1}, t_{m\pm1}, t_b, t_{s\pm1}, t_{g\pm1}$. For those, we use the most

[1] (Li et al., 2011) adopt a complex strategy to incorporate the POS features in their joint decoding algorithm. The way illustrated here is much easier and its correctness can be easily proved.

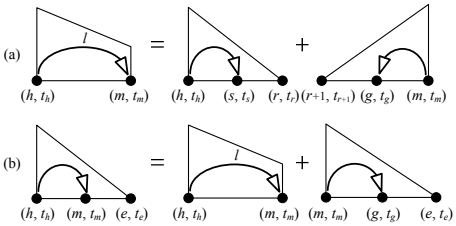


Figure 3: An illustration of the dynamic programming based decoding algorithm for our joint model. We omit the creation of right-headed spans for brevity.

likely POS tags provided by the baseline CRF tagging model following (Li et al., 2011). They find that this approximation substantially improves the efficiency of their joint models without accuracy loss. The time complexity of the algorithm is $O(|\mathscr{L}|n^4q^5)$, and the space complexity is $O(|\mathscr{L}|n^2q^2 + n^3q^3)$ where q is the tag number of each word ($\leq |\mathscr{T}|$).

POS tag pruning. Since the time complexity is high in terms of q, we follow (Li et al., 2011) and prune out the lower-probability POS tags for each word based on their marginal probabilities provided by our baseline CRF model. After pruning, each word has 1.4 candidate POS tags on average and the oracle tagging accuracy is 99.27% on CTB5. On PTB, each word has 1.2 candidate POS tags and the oracle is 99.71%.

Dependency pruning. The parsing time grows quickly with n. Therefore, we train a CRF-based first-order dependency parser to eliminate the unlikely dependencies following (Charniak and Johnson, 2005; Petrov and Klein, 2007; Koo and Collins, 2010). After pruning, 31.3% of the dependencies are left and the oracle dependency accuracy (UAS) is 99.77% on CTB5. On PTB, 28.9% of the dependencies are retained and the oracle UAS is 99.91%.

5 A separately passive-aggressive training algorithm

Online training has proven successful in several structured classification problems such as POS tagging (Collins, 2002) and parsing (McDonald et al., 2005; Zhang and Clark, 2011). Algorithm 1 shows the generic framework of online training when applied to our joint task. Online training iteratively traverses the entire training dataset and use one instance to update the feature weights at each time. First, the best result for the instance is found based on the current feature weights (line 6). Then, the feature weights are updated by comparing the best result and the gold-standard reference (line 7).

According to the update criterion, three different online training algorithms are widely used in parsing community, i.e. *averaged perceptron (AP)* (Collins, 2002), *passive-aggressive algorithm (PA)* (Crammer et al., 2003), and *margin infused relaxed algorithm (MIRA)* (Crammer and Singer, 2001). Previous work mostly adopts AP to train their joint models (Li et al., 2011; Hatori et al., 2011; Bohnet and Nivre, 2012). We compare AP and PA for our joint models and find similar accuracy in both tagging and parsing. Then we propose a variant of PA named as separately passive-aggressive algorithm (SPA) to improve the tagging accuracy. For the sake of conciseness, we do not make comparison with MIRA, since our preliminary results show that MIRA achieves similar performance to AP and PA.

Algorithm 1 Generic online training for joint POS tagging and dependency parsing

1. **Input:** Training Data $\mathbb{D} = \{(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)})\}_{j=1}^{N}$
2. **Output:** $\mathbf{w}_{\text{joint}}$ $(\equiv \mathbf{w}_{\text{pos}\oplus\text{syn}})$
3. **Initialization:** $\mathbf{w}_{\text{joint}}^{(0)} = \mathbf{0}; \mathbf{v} = \mathbf{0}; k = 0$
4. **for** $i = 1$ **to** I **do** // *iterations*
5. **for** $j = 1$ **to** N **do** // *traverse the samples*
6. $(\hat{\mathbf{t}}, \hat{\mathbf{d}}) = \arg\max_{\mathbf{t},\mathbf{d}} \mathbf{w}_{\text{joint}}^{(k)} \cdot \mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \mathbf{t}, \mathbf{d})$ // *decode based on current feature weights.*
7. $\mathbf{w}_{\text{joint}}^{(k+1)} = \text{update } \mathbf{w}_{\text{joint}}^{(k)} \text{ with } (\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \hat{\mathbf{t}}, \mathbf{d}^{(j)}, \hat{\mathbf{d}})$ // *update weights according to some criterion.*
8. $\mathbf{v} = \mathbf{v} + \mathbf{w}_{\text{joint}}^{(k+1)}$
9. $k = k + 1$
10. **end for**
11. **end for**
12. $\mathbf{w}_{\text{joint}} = \mathbf{v}/(I \times N)$ // *average the weights*

All algorithms adopt the update direction of the distance between the reference feature vector $\mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)})$ and the feature vector of the best result $\mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}})$. However, different strategies are adopted to determine the update step. AP uses a constant update step of 1.

$$\texttt{AP}\left\{\mathbf{w}_{\text{joint}}^{(k+1)} = \mathbf{w}_{\text{joint}}^{(k)} + \mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)}) - \mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}})\right. \tag{7}$$

PA computes the update step τ_{joint} by considering the loss of the best result, the score distance, and the feature vector distance.

$$\texttt{PA}\left\{\begin{array}{l}\tau_{\text{joint}} = \dfrac{\text{Score}_{\text{joint}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}}) - \text{Score}_{\text{joint}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)}) + \rho_{\text{pos}}(\mathbf{t}^{(j)}, \hat{\mathbf{t}}) + \rho_{\text{syn}}(\mathbf{d}^{(j)}, \hat{\mathbf{d}})}{\|\mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)}) - \mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}})\|^2} \\ \mathbf{w}_{\text{joint}}^{(k+1)} = \mathbf{w}_{\text{joint}}^{(k)} + \tau_{\text{joint}}(\mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)}) - \mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}}))\end{array}\right. \tag{8}$$

where $\rho_{\text{pos}}(\mathbf{t}^{(j)}, \hat{\mathbf{t}})$ is the incorrect POS tag number in $\hat{\mathbf{t}}$ according to $\mathbf{t}^{(j)}$, and $\rho_{\text{syn}}(\mathbf{d}^{(j)}, \hat{\mathbf{d}})$ is the dependency error number in $\hat{\mathbf{d}}$ according to $\mathbf{d}^{(j)}$. Following (Johansson and Nugues, 2008), $\rho_{\text{syn}}(\mathbf{d}^{(j)}, \hat{\mathbf{d}})$ increases by 1 for an incorrect dependency and by 0.5 for a correct dependency with a wrong label. Theoretically, Eq. 8 computes the smallest update that makes the correct hypothesis outscores the returned highest-scoring hypothesis by the overall error.

We can see that AP and PA use the same update step for the POS features $\mathbf{f}_{\text{pos}}(.)$ and syntactic features $\mathbf{f}_{\text{syn}}(.)$. Therefore, the weights of the POS features and the syntactic features are of the same scale after training is completed. We argue that this is problematic since the number of the syntactic features are much larger than the number of the POS features. We find that in $\mathbf{f}_{\text{joint}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}})$, $\mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}})$ contains more than 3,000 non-zero features, whereas $\mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}})$ only has less than 200 non-zero features on average. As a result, the model is dominated by the syntactic features and the POS features would play a very limited role in determining the best joint result. Actually, we find that the POS features contribute a very small part to the score of the best joint results when trained with AP or PA (see Figure 5).

After several empirical trials, we find that we can raise the score contributed by the POS features by increasing the update step of the POS features. Furthermore, we find that we can elegantly achieve this goal by slightly modifying the update formulas of PA. We name the proposed training algorithm as the *separately passive-aggressive algorithm (SPA)*. SPA separately

computes two update steps for the POS features and the syntactic features.

$$\text{SPA}\begin{cases} \tau_{\text{pos}} = \dfrac{\text{Score}_{\text{pos}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}) - \text{Score}_{\text{pos}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}) + \rho_{\text{pos}}(\mathbf{t}^{(j)}, \hat{\mathbf{t}})}{\|\mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}) - \mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}})\|^2} \\ \tau_{\text{syn}} = \dfrac{\text{Score}_{\text{syn}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}}) - \text{Score}_{\text{syn}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)}) + \rho_{\text{syn}}(\mathbf{d}^{(j)}, \hat{\mathbf{d}})}{\|\mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)}) - \mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}})\|^2} \\ \mathbf{w}_{\text{pos}}^{(k+1)} = \mathbf{w}_{\text{pos}}^{(k)} + \tau_{\text{pos}}(\mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}) - \mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}})) \\ \mathbf{w}_{\text{syn}}^{(k+1)} = \mathbf{w}_{\text{syn}}^{(k)} + \tau_{\text{syn}}(\mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)}) - \mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}})) \end{cases} \quad (9)$$

Analogous to PA, Eq. 9 separately finds the smallest update to the POS feature weights that makes the POS score of the correct hypothesis higher than the POS score of the returned highest-scoring hypothesis by the POS error, and the smallest update to the syntactic feature weights that makes the syntactic score of the correct hypothesis higher than the syntactic score of the returned highest-scoring one by the syntactic error. Note that Eq. 9 sometimes leads to negative τ_{pos} or τ_{syn}, because the correct hypothesis already has a POS or syntactic subscore larger than the 1-best one but the deficiency is entirely confined to the other subscore. However, such cases are rare. We just set the update step to zero when it is negative.

Since $\mathbf{f}_{\text{pos}}(.)$ contains much less non-zero features than $\mathbf{f}_{\text{syn}}(.)$, $\|\mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}) - \mathbf{f}_{\text{pos}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}})\|^2$ is much smaller than $\|\mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{d}^{(j)}) - \mathbf{f}_{\text{syn}}(\mathbf{x}^{(j)}, \hat{\mathbf{t}}, \hat{\mathbf{d}})\|^2$. Therefore, τ_{pos} is much larger than τ_{syn}. Our experiments show that τ_{pos} is about 10 times larger than τ_{syn} on average. This means that the POS features are updated in much larger step than the syntactic features. As a result, the POS features play a more important role in the joint models. We find that the POS score becomes much closer to the syntactic score in the best joint results (see Figure 5).

As discussed in Section 1, (Li et al., 2011) find that their joint models trained with AP are good at resolving syntax-sensitive POS ambiguities like {VV,NN}, whereas their baseline sequential POS tagging model does well in disambiguating the syntax-insensitive ones like {NN,NR}. We believe that it is because the discriminative power of the POS features in resolving such syntax-insensitive POS ambiguities are suppressed in the joint models when trained with AP or PA. Compared with AP and PA, SPA raises the weight of the POS features and can better utilize the disambiguation power of both the POS and syntactic features, leading to large tagging accuracy boost. On the other hand, better tagging results can further help parsing.

6 Experiments

Data. We conduct experiments on CTB5 (Xue et al., 2005). Following the standard practice, we adopt the data split of (Duan et al., 2007; Zhang and Clark, 2008b; Huang and Sagae, 2010) and adopt Penn2Malt[2] for constituent-to-dependency conversion with the head-finding rules of (Zhang and Clark, 2008b).

We also evaluate our models on another version of CTB5 used in (Bohnet and Nivre, 2012) to compare with their joint model. We thank Bernd Bohnet for sharing their dataset. We refer to their dataset as CTB5-Bohnet. We carefully compare CTB5 with CTB5-Bohnet and find that except for the mismatch of about 30 sentence, the datasets differ in both dependency structures and dependency labels. After discussions with Bernd Bohnet, we find out that they adopt Yue Zhang's constituent-to-dependency conversion tool[3] whereas we use Penn2Malt for

[2] http://w3.msi.vxu.se/~nivre/research/Penn2Malt.html
[3] http://sourceforge.net/projects/zpar/files/0.3/

Corpus	Train	Dev	Test
CTB5	16,091	803	1,910
CTB5-Bohnet	16,069	803	1,905
PTB	39,832	1,346	2,416

Table 2: Data used in this work (in sentence number).

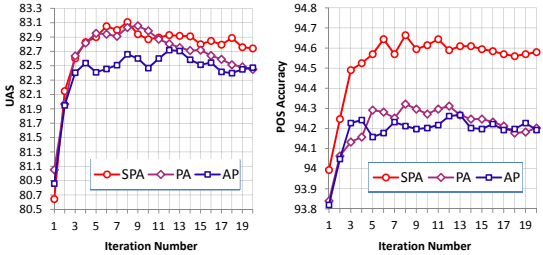


Figure 4: Training curves of UAS and POS tagging accuracy of the unlabeled joint model using SPA/PA/AP on the CTB5 development dataset.

CTB5, although the head-finding rules are the same.

For English, we adopt PTB (sec 02-21 for training, sec 24 for development, sec 23 for test) and convert the original bracketed structures into dependencies using Penn2Malt with its default head-finding rules. Table 2 summarizes the data sets used in the present work.

Evaluation metrics. We use the standard *POS tagging accuracy (POS)* to evaluate POS tagging. For dependency parsing, we use *unlabeled attachment score (UAS)* and *labeled attachment score (LAS)* (all excluding punctuation) (Hajič et al., 2009).

6.1 Experiments on CTB5

6.1.1 Comparison of the three training algorithms

The curves in Figure 4 show the performance of the unlabeled joint model on the development dataset using SPA/PA/AP after each iteration during training. We can see that the PA outperforms AP in both UAS and tagging accuracy. SPA achieves a slightly higher peak UAS than PA, and substantially outperforms both PA and AP in tagging accuracy. In addition, Figure 4 empirically indicates that SPA can converge as fast as AP and PA. We leave the theoretic proof of the convergence of SPA in the future work.

To better understand the reason behind the improvement in tagging accuracy, we try to analyze the two-part model scores, i.e. the POS score $\mathsf{Score}_{\mathrm{pos}}(.)$ and the syntactic score $\mathsf{Score}_{\mathrm{syn}}(.)$. After each iteration, we parse the development dataset using the current weights, and then compute and average each part of the model score. Figure 5 shows the results. For PA and AP, the scale of the POS score is about 1/50 ($10^{1.7} = 50$) of the syntactic score, which means the POS features play an insignificant role in determining the joint result. Obviously, SPA can raise the weight of the POS features, as the POS score is about 1/8 ($10^{0.9} = 8$) of the syntactic score.

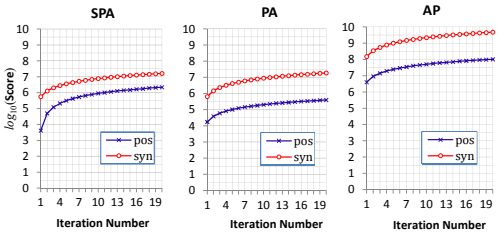


Figure 5: Model score analysis on the CTB5 development dataset. "pos" and "syn" refer to $\text{Score}_{\text{pos}}(.)$ and $\text{Score}_{\text{syn}}(.)$ (see Eq. 6).

	UAS	POS
SPA	81.21	94.51
PA	80.98	94.18
AP	80.94	94.17
pipeline	80.22	93.88

Table 3: Performance of three training algorithms on the CTB5 test dataset.

As a result, the POS features can make more contributions, and the discriminative power of the syntactic features in resolving the syntax-sensitive ambiguities and the power of the POS features in the syntax-insensitive POS ambiguities can be better balanced. We believe this is the reason behind the improvement in tagging accuracy.

Table 3 reports the results of our unlabeled joint model trained with SPA/PA/AP on the CTB5 test dataset. We adopt the parameters which lead to the best parsing accuracy on the development dataset as shown in Figure 4. We also present the performance of our pipelined unlabeled parsing model trained with PA. Different to (Li et al., 2011), our joint models trained with AP or PA both outperform our baseline POS tagging model by about 0.3%. We believe that this may be due to the richer syntactic features incorporated in our joint models. Moreover, SPA can outperform both PA and AP in tagging accuracy by more than 0.3%, which is an error reduction of 5%. Meanwhile, SPA also improves UAS by about 0.2% since better tagging results help parsing. This again demonstrates the effectiveness of the proposed SPA.

6.1.2 Main results

Table 4 shows the final results on the CTB5 test set. Three sets of results are presented. For "gold POS" and "pipeline", we train the parsing models with PA. For "joint" models, we adopt SPA. For each case, we train our models with two different settings, one labeled and one unlabeled, to study the effect of dependency labels. We can see that we achieve best results on all three cases. Specifically, a few interesting conclusions can be drawn.

- Labeling the dependencies during parsing improves UAS by 0.6%/0.5% in the pipeline and joint cases and by 0.2% when using gold-standard POS tags. Dependency labels also slightly help POS tagging in the joint case.
- Compared with the pipeline models, the joint models with SPA can largely improve POS

	Models	LAS	UAS	POS
joint	Ours (labeled)	**79.01**	**81.67**	**94.60**
	Ours (unlabeled)	—	81.21	94.51
	(Li et al., 2011)	—	80.74	93.08
	(Hatori et al., 2011)	—	81.33	93.94
pipeline	Ours (labeled)	**77.80**	**80.82**	**93.88**
	Ours (unlabeled)	—	80.22	
	(Li et al., 2011)	—	79.29	93.51
	(Hatori et al., 2011)	—	78.04	93.82
gold POS	Ours (labeled)	**85.36**	**86.76**	
	Ours (unlabeled)	—	86.55	
	(Li et al., 2011)	—	86.18	100.0
	(Hatori et al., 2011)	—	85.96	
	(Zhang and Nivre, 2011)	84.4	86.0	
	(Huang and Sagae, 2010)	—	85.20	

Table 4: Final results on the CTB5 test dataset.

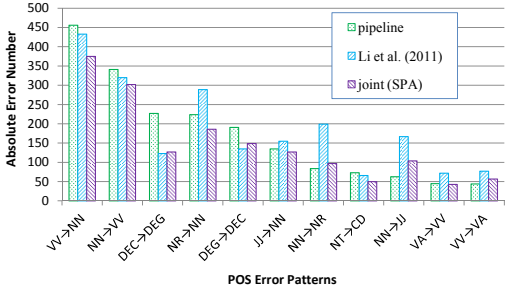


Figure 6: Statistics for different POS error patterns on the CTB5 test dataset.

tagging accuracy by 0.6-0.7%, which is an error reduction of 10%.

- Our joint models largely outperforms their pipeline counterparts by 0.9-1.0% in UAS and 1.2% in LAS.

6.1.3 Analysis

Figure 6 compares different models on a number of high-frequency POS error patterns. An error pattern X→Y means that the focus word, whose true tag is X, is assigned a tag Y. We thank the authors of (Li et al., 2011) for sharing their results. The joint model of (Li et al., 2011) reduces errors for syntax-sensitive ambiguities such as {(DEC,DEG} and {NN,VV}, but largely increases errors for syntax-insensitive ambiguities like {NN,NR}, {NN,JJ}, and {VV,VA}, which can explain its low tagging accuracy. Compared with (Li et al., 2011), our joint model trained with SPA does much better in resolving the syntax-insensitive ambiguities and achieves similar performance to the baseline CRF-based tagging model on those patterns. In summary, we can conclude that our joint model trained with SPA performs similarly to the baseline CRF-

Models		LAS	UAS	POS
joint	Ours (labeled)	**80.18**	**83.14**	**94.71**
	Ours (unlabeled)	—	82.37	94.65
	(Bohnet and Nivre, 2012)	77.91	81.42	93.24
pipeline	Ours (labeled)	**79.01**	**82.07**	**93.89**
	Ours (unlabeled)	—	81.38	
	(Bohnet and Nivre, 2012)	76.79	80.33	92.81

Table 5: Final results on the CTB5-Bohnet test dataset.

Models		LAS	UAS	POS
joint	Ours (labeled)	**92.44**	**93.52**	**97.62**
	Ours (unlabeled)	—	93.12	**97.62**
	(Bohnet and Nivre, 2012)	**92.44**	93.38	97.33
	(Bohnet and Nivre, 2012) †	*92.68*	*93.67*	*97.42*
pipeline	Ours (labeled)	**92.00**	93.14	97.16
	Ours (unlabeled)	—	92.85	
	(Bohnet and Nivre, 2012)	91.71	92.79	**97.28**
	(Zhang and Nivre, 2011)	—	92.9	—
	(Martins et al., 2010)	—	**93.26**	—
	(Koo and Collins, 2010)	—	93.04	—
	(Huang and Sagae, 2010)	—	92.1	—
	(Koo et al., 2008) †	—	*93.16*	—
	(Carreras et al., 2008) †	—	*93.5*	—
	(Suzuki et al., 2009) †	—	*93.79*	—

Table 6: Final results on the PTB test dataset. Results marked by † use additional resources and are therefore not directly comparable to the others.

based tagging model on the syntax-insensitive ambiguities and meanwhile similarly to the joint model of (Li et al., 2011) on the syntax-sensitive ambiguities. This demonstrates that SPA can better balance the discriminative power of both the POS and syntactic features.

6.2 Experiments on CTB5-Bohnet

We conduct experiments on CTB5-Bohnet to make comparison with the recent results of (Bohnet and Nivre, 2012). Table 5 presents the results. We can see that our pipeline and joint models largely outperform those of (Bohnet and Nivre, 2012). Moreover, the parsing accuracies on this dataset are much higher than those on CTB5 due to the different conversion strategy. We will study the reasons in the future work. Again, we find that labeled parsing can largely improve UAS.

6.3 Experiments on PTB

To find out the effect of the joint models on English, we conduct experiments on PTB. Table 6 shows the results. Several state-of-the-art results are also presented. The pipeline models on English suffer from less error propagation problem than that on Chinese, as the POS tag-

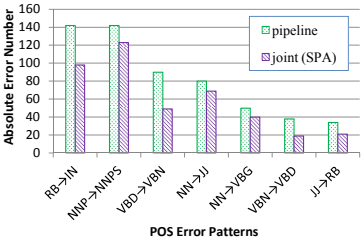


Figure 7: Statistics for different POS error patterns on the PTB test dataset.

ging accuracy on English is much higher (97% vs. 94%). However, we still find that our joint models can obtain a 0.3-0.4% gain in UAS, a 0.4% gain in LAS, and a 0.5% gain in tagging accuracy. Similar to the findings on the Chinese datasets, we demonstrate that labeled parsing can boost UAS by 0.3-0.4% on English. Our labeled joint model outperforms the one of (Bohnet and Nivre, 2012) by 0.1% in UAS and 0.3% in tagging accuracy. Actually, our labeled joint model achieves the best parsing and tagging accuracy on PTB.

We compare the POS tagging results of the unlabeled joint model and the baseline tagging model. Figure 7 shows a few high-frequency error patterns. We can see that the joint model can help resolve a variety of POS ambiguities. For example, the error number of RB→IN (adverbs wrongly tagged as prepositions or subordinating conjunctions) are reduced from 142 to 98, which is a 31% error reduction. Also, the ambiguous pair {VBD, VBN} (verbs of past tense, verbs of past participle) are much better disambiguated by the joint model.

Conclusions

This paper presents a separately passive-aggressive training algorithm (SPA) for joint POS tagging and labeled dependency parsing models. We show that SPA can more properly learn the feature weights than the averaged perceptron (AP) and the passive aggressive algorithm (PA) and can better balance the discriminative power of the POS feature in resolving the syntax-insensitive POS ambiguities and the power of the syntactic features in resolving the syntax-sensitive ambiguities, leading to large tagging accuracy improvement. On the other hand, better POS tagging can further help parsing.

For future work, we are interested in studying SPA from the theoretic perspective and try to provide more insights and justifications of its effectiveness in training the joint models. Besides, although this paper focuses on graph-based joint models, we believe that SPA can also be applied to transition-based joint models (Hatori et al., 2011; Bohnet and Nivre, 2012).

Acknowledgements

We thank Meishan Zhang, for suggesting the easier way to incorporate the POS features during joint decoding, and the anonymous reviewers, for their valuable comments which lead to better understanding of the proposed SPA. This work was supported by National Natural Science Foundation of China (NSFC) via grant 61133012, the National “863” Major Projects via grant 2011AA01A207, and the National “863” Leading Technology Research Project via grant 2012AA011102.

References

Auli, M. and Lopez, A. (2011). A comparison of loopy belief propagation and dual decomposition for integrated ccg supertagging and parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 470–480, Portland, Oregon, USA. Association for Computational Linguistics.

Bohnet, B. (2010). Top accuracy and fast dependency parsing is not a contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 89–97, Beijing, China. Coling 2010 Organizing Committee.

Bohnet, B. and Nivre, J. (2012). A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1455–1465, Jeju Island, Korea. Association for Computational Linguistics.

Carreras, X. (2007). Experiments with a higher-order projective dependency parser. In *Proceedings of EMNLP/CoNLL*, pages 141–150.

Carreras, X., Collins, M., and Koo, T. (2008). Tag, dynamic programming, and the perceptron for efficient, feature-rich parsing. In *CoNLL 2008: Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 9–16, Manchester, England. Coling 2008 Organizing Committee.

Charniak, E. and Johnson, M. (2005). Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of ACL-05*, pages 173–180.

Che, W., Spitkovsky, V., and Liu, T. (2012). A comparison of chinese parsers for stanford dependencies. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 11–16, Jeju Island, Korea. Association for Computational Linguistics.

Collins, M. (2002). Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of EMNLP 2002*.

Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., and Singer, Y. (2003). Online passive aggressive algorithms. In *Procedings of NIPS 2003*.

Crammer, K. and Singer, Y. (2001). Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 3.

Duan, X., Zhao, J., , and Xu, B. (2007). Probabilistic models for action-based Chinese dependency parsing. In *Proceedings of ECML/ECPPKDD*.

Eisner, J. (2000). Bilexical grammars and their cubic-time parsing algorithms. In *Advances in Probabilistic and Other Parsing Technologies*, pages 29–62.

Hajič, J., Ciaramita, M., Johansson, R., Kawahara, D., Martí, M. A., Màrquez, L., Meyers, A., Nivre, J., Padó, S., Štěpánek, J., Straňák, P., Surdeanu, M., Xue, N., and Zhang, Y. (2009). The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of CoNLL 2009*.

Hatori, J., Matsuzaki, T., Miyao, Y., and Tsujii, J. (2011). Incremental joint pos tagging and dependency parsing in chinese. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1216–1224, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Huang, L. and Sagae, K. (2010). Dynamic programming for linear-time incremental parsing. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1077–1086, Uppsala, Sweden. Association for Computational Linguistics.

Johansson, R. and Nugues, P. (2008). Dependency-based semantic role labeling of PropBank. In *EMNLP-2008*.

Koo, T., Carreras, X., and Collins, M. (2008). Simple semi-supervised dependency parsing. In *Proceedings of ACL-08: HLT*, pages 595–603, Columbus, Ohio. Association for Computational Linguistics.

Koo, T. and Collins, M. (2010). Efficient third-order dependency parsers. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1–11, Uppsala, Sweden. Association for Computational Linguistics.

Lafferty, J., McCallum, A., and Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML 2001*, pages 282–289.

Lee, J., Naradowsky, J., and Smith, D. A. (2011). A discriminative model for joint morphological disambiguation and dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 885–894, Portland, Oregon, USA. Association for Computational Linguistics.

Li, Z., Zhang, M., Che, W., Liu, T., Chen, W., and Li, H. (2011). Joint models for chinese pos tagging and dependency parsing. In *EMNLP 2011*, pages 1180–1191.

Martins, A., Smith, N., Xing, E., Aguiar, P., and Figueiredo, M. (2010). Turbo parsers: Dependency parsing by approximate variational inference. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 34–44, Cambridge, MA. Association for Computational Linguistics.

McDonald, R., Crammer, K., and Pereira, F. (2005). Online large-margin training of dependency parsers. In *Proceedings of ACL 2005*, pages 91–98.

Petrov, S. and Klein, D. (2007). Improved inference for unlexicalized parsing. In *Proceedings of NAACL 2007*.

Ratnaparkhi, A. (1996). A maximum entropy model for part-of-speech tagging. In *Proceedings of EMNLP 1996*.

Rush, A. M., Sontag, D., Collins, M., and Jaakkola, T. (2010). On dual decomposition and linear programming relaxations for natural language processing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1–11.

Smith, D. A. and Eisner, J. (2008). Dependency parsing by belief propagation. In *Proceedings of EMNLP 2008*, pages 145–156.

Suzuki, J., Isozaki, H., Carreras, X., and Collins, M. (2009). An empirical study of semi-supervised structured conditional models for dependency parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 551–560, Singapore. Association for Computational Linguistics.

Xue, N., Xia, F., Chiou, F.-D., and Palmer, M. (2005). The Penn Chinese Treebank: Phrase structure annotation of a large corpus. In *Natural Language Engineering*, volume 11, pages 207–238.

Zhang, Y. and Clark, S. (2008a). Joint word segmentation and POS tagging using a single perceptron. In *Proceedings of ACL-08: HLT*, pages 888–896.

Zhang, Y. and Clark, S. (2008b). A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 562–571, Honolulu, Hawaii. Association for Computational Linguistics.

Zhang, Y. and Clark, S. (2011). Syntactic processing using the generalized perceptron and beam search. *Computational Linguistics*, 37(1):105ÍC151.

Zhang, Y. and Nivre, J. (2011). Transition-based dependency parsing with rich non-local features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 188–193, Portland, Oregon, USA. Association for Computational Linguistics.