Incremental Joint Approach to Word Segmentation, POS Tagging, and Dependency Parsing in Chinese

Jun Hatori[1] **Takuya Matsuzaki**[2] **Yusuke Miyao**[2] **Jun'ichi Tsujii**[3]
[1]University of Tokyo / 7-3-1 Hongo, Bunkyo, Tokyo, Japan
[2]National Institute of Informatics / 2-1-2 Hitotsubashi, Chiyoda, Tokyo, Japan
[3]Microsoft Research Asia / 5 Danling Street, Haidian District, Beijing, P.R. China
hatori@is.s.u-tokyo.ac.jp
{takuya-matsuzaki,yusuke}@nii.ac.jp jtsujii@microsoft.com

Abstract

We propose the first joint model for word segmentation, POS tagging, and dependency parsing for Chinese. Based on an extension of the incremental joint model for POS tagging and dependency parsing (Hatori et al., 2011), we propose an efficient character-based decoding method that can combine features from state-of-the-art segmentation, POS tagging, and dependency parsing models. We also describe our method to align comparable states in the beam, and how we can combine features of different characteristics in our incremental framework. In experiments using the Chinese Treebank (CTB), we show that the accuracies of the three tasks can be improved significantly over the baseline models, particularly by 0.6% for POS tagging and 2.4% for dependency parsing. We also perform comparison experiments with the partially joint models.

1 Introduction

In processing natural languages that do not include delimiters (e.g. spaces) between words, word segmentation is the crucial first step that is necessary to perform virtually all NLP tasks. Furthermore, the word-level information is often augmented with the POS tags, which, along with segmentation, form the basic foundation of statistical NLP.

Because the tasks of word segmentation and POS tagging have strong interactions, many studies have been devoted to the task of joint word segmentation and POS tagging for languages such as Chinese (e.g. Kruengkrai et al. (2009)). This is because some of the segmentation ambiguities cannot be resolved without considering the surrounding grammatical constructions encoded in a sequence of POS tags. The joint approach to word segmentation and POS tagging has been reported to improve word segmentation and POS tagging accuracies by more than 1% in Chinese (Zhang and Clark, 2008). In addition, some researchers recently proposed a joint approach to Chinese POS tagging and dependency parsing (Li et al., 2011; Hatori et al., 2011); particularly, Hatori et al. (2011) proposed an incremental approach to this joint task, and showed that the joint approach improves the accuracies of these two tasks.

In this context, it is natural to consider further a question regarding the joint framework: how strongly do the tasks of word segmentation and dependency parsing interact? In the following Chinese sentences:

当今 和平奖 与 和平 事业 相关
current peace-prize and peace operation related
The current peace prize and peace operations are related.

当今 和平 奖与 和平 事业 相关 团体
current peace award peace operation related group
The current peace is awarded to peace-operation-related groups.

the only difference is the existence of the last word 团体; however, whether or not this word exists changes the whole syntactic structure and segmentation of the sentence. This is an example in which word segmentation cannot be handled properly without considering long-range syntactic information.

Syntactic information is also considered beneficial to improve the segmentation of out-of-vocabulary (OOV) words. Unlike languages such as Japanese that use a distinct character set (i.e. *katakana*) for foreign words, the transliterated words in Chinese, many of which are OOV words, frequently include characters that are also used as common or function words. In the current systems, the existence of these characters causes numerous over-segmentation errors for OOV words.

Based on these observations, we aim at building a joint model that simultaneously processes word segmentation, POS tagging, and dependency parsing, trying to capture global interaction among

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 1045–1053,
Jeju, Republic of Korea, 8-14 July 2012.

these three tasks. To handle the increased computational complexity, we adopt the incremental parsing framework with dynamic programming (Huang and Sagae, 2010), and propose an efficient method of character-based decoding over candidate structures.

Two major challenges exist in formalizing the joint segmentation and dependency parsing task in the character-based incremental framework. First, we must address the problem of how to align comparable states effectively in the beam. Because the number of dependency arcs varies depending on how words are segmented, we devise a step alignment scheme using the number of character-based arcs, which enables effective joint decoding for the three tasks.

Second, although the feature set is fundamentally a combination of those used in previous works (Zhang and Clark, 2010; Huang and Sagae, 2010), to integrate them in a single incremental framework is not straightforward. Because we must perform decisions of three kinds (segmentation, tagging, and parsing) in an incremental framework, we must adjust which features are to be activated when, and how they are combined with which action labels. We have also found that we must balance the learning rate between features for segmentation and tagging decisions, and those for dependency parsing.

We perform experiments using the Chinese Treebank (CTB) corpora, demonstrating that the accuracies of the three tasks can be improved significantly over the pipeline combination of the state-of-the-art joint segmentation and POS tagging model, and the dependency parser. We also perform comparison experiments with partially joint models, and investigate the tradeoff between the running speed and the model performance.

2 Related Works

In Chinese, Luo (2003) proposed a joint constituency parser that performs segmentation, POS tagging, and parsing within a single character-based framework. They reported that the POS tags contribute to segmentation accuracies by more than 1%, but the syntactic information has no substantial effect on the segmentation accuracies. In contrast, we built a joint model based on a dependency-based framework, with a rich set of structural features. Using it, we show the first positive result in Chinese that the segmentation accuracies can be improved using the syntactic information.

Another line of work exists on lattice-based parsing for Semitic languages (Cohen and Smith, 2007; Goldberg and Tsarfaty, 2008). These methods first convert an input sentence into a lattice encoding the morphological ambiguities, and then conduct joint morphological segmentation and PCFG parsing. However, the segmentation possibilities considered in those studies are limited to those output by an existing morphological analyzer. In addition, the lattice does not include word segmentation ambiguities crossing boundaries of space-delimited tokens. In contrast, because the Chinese language does not have spaces between words, we fundamentally need to consider the lattice structure of the whole sentence. Therefore, we place no restriction on the segmentation possibilities to consider, and we assess the full potential of the joint segmentation and dependency parsing model.

Among the many recent works on joint segmentation and POS tagging for Chinese, the linear-time incremental models by Zhang and Clark (2008) and Zhang and Clark (2010) largely inspired our model. Zhang and Clark (2008) proposed an incremental joint segmentation and POS tagging model, with an effective feature set for Chinese. However, it requires to computationally expensive multiple beams to compare words of different lengths using beam search. More recently, Zhang and Clark (2010) proposed an efficient character-based decoder for their word-based model. In their new model, a single beam suffices for decoding; hence, they reported that their model is practically ten times as fast as their original model. To incorporate the word-level features into the character-based decoder, the features are decomposed into substring-level features, which are effective for incomplete words to have comparable scores to complete words in the beam. Because we found that even an incremental approach with beam search is intractable if we perform the word-based decoding, we take a character-based approach to produce our joint model.

The incremental framework of our model is based on the joint POS tagging and dependency parsing model for Chinese (Hatori et al., 2011), which is an extension of the shift-reduce dependency parser with dynamic programming (Huang and Sagae, 2010). They specifically modified the shift action so that it assigns the POS tag when a word is shifted onto the stack. However, because they regarded word segmentation as given, their model did not consider the

interaction between segmentation and POS tagging.

3 Model

3.1 Incremental Joint Segmentation, POS Tagging, and Dependency Parsing

Based on the joint POS tagging and dependency parsing model by Hatori et al. (2011), we build our joint model to solve word segmentation, POS tagging, and dependency parsing within a single framework. Particularly, we change the role of the shift action and additionally use the append action, inspired by the character-based actions used in the joint segmentation and POS tagging model by Zhang and Clark (2010).

The list of actions used is the following:

- A: *append* the first character in the queue to the word on top of the stack.
- SH(t): *shift* the first character in the input queue as a new word onto the stack, with POS tag t.
- RL/RR: *reduce* the top two trees on the stack, (s_0, s_1), into a subtree $s_0 \curvearrowright s_1$ / $s_0 \curvearrowleft s_1$, respectively.

Although SH(t) is similar to the one used in Hatori et al. (2011), now it shifts the first character in the queue as a new word, instead of shifting a word. Following Zhang and Clark (2010), the POS tag is assigned to the word when its first character is shifted, and the word–tag pairs observed in the training data and the closed-set tags (Xia, 2000) are used to prune unlikely derivations. Because 33 tags are defined in the CTB tag set (Xia, 2000), our model exploits a total of 36 actions.

To train the model, we use the averaged perceptron with the early update (Collins and Roark, 2004). In our joint model, the early update is invoked by mistakes in any of word segmentation, POS tagging, or dependency parsing.

3.2 Alignment of States

When dependency parsing is integrated into the task of joint word segmentation and POS tagging, it is not straightforward to define a scheme to *align* (*synchronize*) the states in the beam. In beam search, we use the *step index* that is associated with each state: the parser states in process are aligned according to the index, and the beam search pruning is applied to those states with the same index. Consequently, for the beam search to function effectively, all states with the same index must be *comparable*, and all terminal states should have the same step index.

We can first think of using the number of shifted characters as the step index, as Zhang and Clark (2010) does. However, because RL/RR actions can be performed without incrementing the step index, the decoder tends to prefer states with more dependency arcs, resulting more likely in premature choice of 'reduce' actions or oversegmentation of words. Alternatively, we can consider using the number of actions that have been applied as the step index, as Hatori et al. (2011) does. However, this results in inconsistent numbers of actions to reach the terminal states: some states that segment words into larger chunks reach a terminal state earlier than other states with smaller chunks. For these reasons, we have found that both approaches yield poor models that are not at all competitive with the baseline (pipeline) models[1].

To address this issue, we propose an indexing scheme using the number of character-based arcs. We presume that in addition to the word-to-word dependency arcs, each word (of length M) implicitly has $M-1$ inter-character arcs, as in: $\boxed{\text{A}\curvearrowright\text{B}\curvearrowright\text{C}}$, $\boxed{\text{A}\curvearrowright\text{B}}\curvearrowright\boxed{\text{C}}$, and $\boxed{\text{A}}\curvearrowright\boxed{\text{B}}\curvearrowright\boxed{\text{C}}$ (each rectangle denotes a word). Then we can define the step index as the sum of the number of shifted characters and the total number of (inter-word and intra-word) dependency arcs, which thereby meets all the following conditions:

(1) All subtrees spanning M consecutive characters have the same index $2M-1$.
(2) All terminal states have the same step index $2N$ (including the root arc), where N is the number of characters in the sentence.
(3) Every action increases the index.

Note that the number of shifted characters is also necessary to meet condition (3). Otherwise, it allows an unlimited number of SH(t) actions without incrementing the step index. Figure 1 portrays how the states are aligned using the proposed scheme, where a subtree is denoted as a rectangle with its partial index shown inside it.

In our framework, because an action increases the step index by 1 (for SH(t) or RL/RR) or 2 (for A), we need to use two beams to store new states at each step. The computational complexity of the entire process is $\mathrm{O}(B(T+3)\cdot 2N)$, where B is the beam

[1] For example, in our preliminary experiment on CTB-5, the step indexing according to the number of actions underperforms the baseline model by 0.2–0.3% in segmentation accuracy.

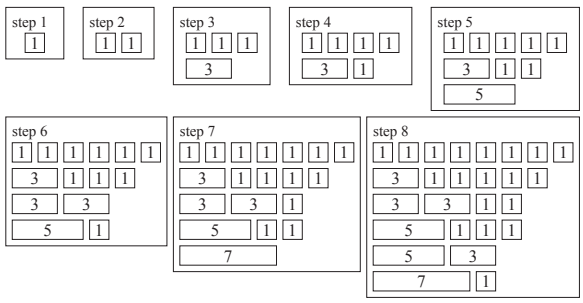

Figure 1: Illustration of the alignment of steps.

size, T is the number of POS tags ($= 33$), and N is the number of characters in the sentence. Theoretically, the computational time is greater than that with the character-based joint segmentation and tagging model by Zhang and Clark (2010) by a factor of $\frac{T+3}{T+1} \cdot \frac{2N}{N} \simeq 2.1$, when the same beam size is used.

3.3 Features

The feature set of our model is fundamentally a combination of the features used in the state-of-the-art joint segmentation and POS tagging model (Zhang and Clark, 2010) and dependency parser (Huang and Sagae, 2010), both of which are used as baseline models in our experiment. However, we must carefully adjust which features are to be activated and when, and how they are combined with which action labels, depending on the type of the features because we intend to perform three tasks in a single incremental framework.

The list of the features used in our joint model is presented in Table 1, where S01–S05, W01–W21, and T01–05 are taken from Zhang and Clark (2010), and P01–P28 are taken from Huang and Sagae (2010). Note that not all features are always considered: each feature is only considered if the action to be performed is included in the list of actions in the "When to apply" column. Because S01–S05 are used to represent the likelihood score of substring sequences, they are only used for A and SH(t) without being combined with any action label. Because T01–T05 are used to determine the POS tag of the word being shifted, they are only applied for SH(t). Because W01–W21 are used to determine whether to segment at the current position or not, they are only used for those actions involved in boundary determination decisions (A, SH(t), RL_0, and RR_0). The action labels RL_0/RR_0 are used to denote the 'reduce' actions that determine the word boundary[2], whereas RL_1/RR_1 denote those 'reduce' actions that are applied when the word boundary has already been fixed. In addition, to capture the shared nature of boundary determination actions (SH(t), RL_0/RR_0), we use a generalized action label SH' to represent any of them when combined with W01–W21. We also propose to use the features U01–U03, which we found are effective to adjust the character-level and substring-level scores.

Regarding the parsing features P01–P28, because we found that P01–P17 are also useful for segmentation decisions, these features are applied to all actions including A, with an explicit distinction of action labels RL_0/RR_0 from RL_1/RR_1. On the other hand, P18–P28 are only used when one of the parser actions (SH(t), RL, or RR) is applied. Note that P07–P09 and P18–P21 (*look-ahead features*) require the look-ahead information of the next word form and POS tags, which cannot be incorporated straightforwardly in an incremental framework. Although we have found that these features can be incorporated using the *delayed features* proposed by Hatori et al. (2011), we did not use them in our current model because it results in the significant increase of computational time.

3.3.1 Dictionary features

Because segmentation using a dictionary alone can serve as a strong baseline in Chinese word segmentation (Sproat et al., 1996), the use of dictionaries is expected to make our joint model more robust and enables us to investigate the contribution of the syntactic dependency in a more realistic setting. Therefore, we optionally use four features D01–D04 associated with external dictionaries. These features distinguish each dictionary source, reflecting the fact that different dictionaries have different characteristics. These features will also be used in our reimplementation of the model by Zhang and Clark (2010).

3.4 Adjusting the Learning Rate of Features

In formulating the three tasks in the incremental framework, we found that adjusting the update rate depending on the type of the features (segmentation/tagging vs. parsing) crucially impacts the final performance of the model. To investigate this point, we define the feature vector $\vec{\phi}$ and score Φ of the

[2] A reduce action has an additional effect of fixing the boundary of the top word on the stack if the last action was A or SH(t).

Id	Feature template	Label	When to apply
U01	$q_{-1}.e \circ q_{-1}.t$	ϕ	A, SH(t)
U02,03	$q_{-1}.e \quad q_{-1}.e \circ q_{-1}.t$	as-is	any
S01	$q_{-1}.e \circ c_0$	ϕ	A
S02	$q_{-1}.t \circ c_0$	ϕ	A, SH(t)
S03	$q_{-1}.t \circ q_{-1}.b \circ c_0$	ϕ	A
S04	$q_{-1}.t \circ c_0 \circ \mathcal{C}(q_{-1}.b)$	ϕ	A
S05	$q_{-1}.t \circ c_0 \circ c_1$	ϕ	A
D01	$\mathrm{len}(q_{-1}.w) \circ i$	A,SH'	A, SH(t), RR/RL$_0$
D02	$\mathrm{len}(q_{-1}.w) \circ q_{-1}.t \circ i$	A,SH'	A, SH(t), RR/RL$_0$
D03	$\mathrm{len}(q_{-1}.w) \circ i$	A,SH'	A, SH(t), RR/RL$_0$
D04	$\mathrm{len}(q_{-1}.w) \circ q_{-1}.t \circ i$	A,SH'	A, SH(t), RR/RL$_0$
	(D01,02: if $q_{-1}.w \in \mathcal{D}_i$; D03,04: if $q_{-1}.w \notin \mathcal{D}_i$)		
W01,02	$q_{-1}.w \quad q_{-2}.w \circ q_{-1}.w$	A,SH'	A, SH(t), RR/RL$_0$
W03	$q_{-1}.w$ (for single-char word)	A,SH'	A, SH(t), RR/RL$_0$
W04	$q_{-1}.b \circ \mathrm{len}(q_{-1}.w)$	A,SH'	A, SH(t), RR/RL$_0$
W05	$q_{-1}.e \circ \mathrm{len}(q_{-1}.w)$	A,SH'	A, SH(t), RR/RL$_0$
W06,07	$q_{-1}.e \circ c_0 \quad q_{-1}.b \circ q_{-1}.e$	A,SH'	A, SH(t), RR/RL$_0$
W08,09	$q_{-1}.w \circ c_0 \quad q_{-2}.e \circ q_{-1}.w$	A,SH'	A, SH(t), RR/RL$_0$
W10,11	$q_{-1}.b \circ c_0 \quad q_{-2}.e \circ q_{-1}.e$	A,SH'	A, SH(t), RR/RL$_0$
W12	$q_{-2}.w \circ \mathrm{len}(q_{-1}.w)$	A,SH'	A, SH(t), RR/RL$_0$
W13	$\mathrm{len}(q_{-2}.w) \circ q_{-1}.w$	A,SH'	A, SH(t), RR/RL$_0$
W14	$q_{-1}.w \circ q_{-1}.t$	A,SH'	A, SH(t), RR/RL$_0$
W15	$q_{-2}.t \circ q_{-1}.w$	A,SH'	A, SH(t), RR/RL$_0$
W16	$q_{-1}.t \circ q_{-1}.w \circ q_{-2}.e$	A,SH'	A, SH(t), RR/RL$_0$
W17	$q_{-1}.t \circ q_{-1}.w \circ c_0$	A,SH'	A, SH(t), RR/RL$_0$
W18	$q_{-2}.e \circ q_{-1}.w \circ c_0 \circ q_1.t$	A,SH'	A, SH(t), RR/RL$_0$
W19	$q_{-1}.t \circ q_{-1}.e$	A,SH'	A, SH(t), RR/RL$_0$
W20	$q_{-1}.t \circ q_{-1}.e \circ c$	A,SH'	A, SH(t), RR/RL$_0$
W21	$q_{-1}.t \circ c \circ \mathrm{cat}(q_{-1}.e)$	A,SH'	A, SH(t), RR/RL$_0$
	(W20, W21: $c \in q_{-1}.w \backslash e$)		
T01,02	$q_{-1}.t \quad q_{-2}.t \circ q_{-1}.t$	SH(t)	SH(t)
T03,04	$q_{-1}.w \quad c_0$	SH(t)	SH(t)
T05	$c_0 \circ q_{-1}.t \circ q_{-1}.e$	SH(t)	SH(t)
P01,02	$s_0.w \quad s_0.t$	A, SH(t), RR/RL$_{0/1}$	any
P03,04	$s_0.w \circ s_0.t \quad s_1.w$	A, SH(t), RR/RL$_{0/1}$	any
P05,06	$s_1.t \quad s_1.w \circ s_1.t$	A, SH(t), RR/RL$_{0/1}$	any
P07,08	$q_0.w \quad q_0.t$	A, SH(t), RR/RL$_{0/1}$	any
P09,10	$q_0.w \circ q_0.t \quad s_0.w \circ s_1.w$	A, SH(t), RR/RL$_{0/1}$	any
P11,12	$s_0.t \circ s_1.t \quad s_0.t \circ q_0.t$	A, SH(t), RR/RL$_{0/1}$	any
P13	$s_0.w \circ s_0.t \circ s_1.t$	A, SH(t), RR/RL$_{0/1}$	any
P14	$s_0.t \circ s_1.w \circ s_1.t$	A, SH(t), RR/RL$_{0/1}$	any
P15	$s_0.w \circ s_1.w \circ s_1.t$	A, SH(t), RR/RL$_{0/1}$	any
P16	$s_0.w \circ s_0.t \circ s_1.w$	A, SH(t), RR/RL$_{0/1}$	any
P17	$s_0.w \circ s_0.t \circ s_1.w \circ s_1.t$	A, SH(t), RR/RL$_{0/1}$	any
P18	$s_0.t \circ q_0.t \circ q_1.t$	as-is	SH(t), RR, RL
P19	$s_1.t \circ s_0.t \circ q_0.t$	as-is	SH(t), RR, RL
P20	$s_0.w \circ q_0.t \circ q_1.t$	as-is	SH(t), RR, RL
P21	$s_1.t \circ s_0.w \circ q_0.t$	as-is	SH(t), RR, RL
P22	$s_1.t \circ s_1.\mathrm{rc}.t \circ s_0.t$	as-is	SH(t), RR, RL
P23	$s_1.t \circ s_1.\mathrm{lc}.t \circ s_0.t$	as-is	SH(t), RR, RL
P24	$s_1.t \circ s_1.\mathrm{rc}.t \circ s_0.w$	as-is	SH(t), RR, RL
P25	$s_1.t \circ s_1.\mathrm{lc}.t \circ s_0.w$	as-is	SH(t), RR, RL
P26	$s_1.t \circ s_0.t \circ s_0.\mathrm{rc}.t$	as-is	SH(t), RR, RL
P27	$s_1.t \circ s_0.w \circ s_0.\mathrm{lc}.t$	as-is	SH(t), RR, RL
P28	$s_2.t \circ s_1.t \circ s_0.t$	as-is	SH(t), RR, RL

* q_{-1} and q_{-2} respectively denote the last-shifted word and the word shifted before q_{-1}. $q.w$ and $q.t$ respectively denote the (root) word form and POS tag of a subtree (word) q, and $q.b$ and $q.e$ the beginning and ending characters of $q.w$. c_0 and c_1 are the first and second characters in the queue. $q.w\backslash e$ denotes the set of characters excluding the ending character of $q.w$. $\mathrm{len}(\cdot)$ denotes the length of the word, capped at 16 if longer. $\mathrm{cat}(\cdot)$ denotes the category of the character, which is the set of POS tags observed in the training data. $\mathcal{D}_i$ is a dictionary, a set of words. The action label ϕ means that the feature is not combined with any label; "as-is" denotes the use of the default action set "A, SH(t), and RR/RL" as is.

Table 1: Feature templates for the full joint model.

	Training		Development			Test		
	#snt	#wrd	#snt	#wrd	#oov	#snt	#wrd	#oov
CTB-5d	16k	438k	804	21k	1.2k	1.9k	50k	3.1k
CTB-5j	18k	494k	352	6.8k	553	348	8.0k	278
CTB-5c	15k	423k	-	-	-	-	-	-
CTB-6	23k	641k	2.1k	60k	3.3k	2.8k	82k	4.6k
CTB-7	31k	718k	10k	237k	13k	10k	245k	13k

Table 2: Statistics of datasets.

action a being applied to the state ψ as

$$\Phi(\psi, a) = \vec{\lambda} \cdot \vec{\phi}(\psi, a) = \vec{\lambda} \cdot \left\{ \vec{\phi}_{st}(\psi, a) + \sigma_p \vec{\phi}_p(\psi, a) \right\},$$

where $\vec{\phi}_{st}$ corresponds to the segmentation and tagging features (those starting with 'U', 'S', 'T', or 'D'), and $\vec{\phi}_p$ is the set of the parsing features (starting with 'P'). Then, if we set σ_p to a number smaller than 1, perceptron updates for the parsing features will be kept small at the early stage of training because the update is proportional to the values of the feature vector. However, even if σ_p is initially small, the global weights for the parsing features will increase as needed and compensate for the small σ_p as the training proceeds. In this way, we can control the contribution of syntactic dependencies at the early stage of training. Section 4.3 shows that the best setting we found is $\sigma_p = 0.5$: this result suggests that we probably should resolve remaining errors by preferentially using the local n-gram based features at the early stage of training. Otherwise, the premature incorporation of the non-local syntactic dependencies might engender overfitting to the training data.

4 Experiment

4.1 Experimental Settings

We use the Chinese Penn Treebank ver. 5.1, 6.0, and 7.0 (hereinafter CTB-5, CTB-6, and CTB-7) for evaluation. These corpora are split into training, development, and test sets, according to previous works. For CTB-5, we refer to the split by Duan et al. (2007) as CTB-5d, and to the split by Jiang et al. (2008) as CTB-5j. We also prepare a dataset for cross validation: the dataset CTB-5c consists of sentences from CTB-5 excluding the development and test sets of CTB-5d and CTB-5j. We split CTB-5c into five sets (CTB-5c-n), and alternatively use four of these as the training set and the rest as the test set. CTB-6 is split according to the official split

described in the documentation, and CTB-7 is split according to Wang et al. (2011). The statistics of these splits are shown in Table 2. As external dictionaries, we use the HowNet Word List[3], consisting of 91,015 words, and page names from the Chinese Wikipedia[4] as of Oct 26, 2011, consisting of 709,352 words. These dictionaries only consist of word forms with no frequency or POS information.

We use standard measures of word-level precision, recall, and F1 score, for evaluating each task. The output of dependencies cannot be correct unless the syntactic head and dependent of the dependency relation are both segmented correctly. Following the standard setting in dependency parsing works, we evaluate the task of dependency parsing with the unlabeled attachment scores excluding punctuations. Statistical significance is tested by McNemar's test ($\dagger : p < 0.05, \ddagger : p < 0.01$).

4.2 Baseline and Proposed Models

We use the following baseline and proposed models for evaluation.

- **SegTag**: our reimplementation of the joint segmentation and POS tagging model by Zhang and Clark (2010). Table 5 shows that this reimplementation almost reproduces the accuracy of their implementation. We used the beam of 16, which they reported to achieve the best accuracies.
- **Dep'**: the state-of-the-art dependency parser by Huang and Sagae (2010). We used our reimplementation, which is used in Hatori et al. (2011).
- **Dep**: Dep' without look-ahead features.
- **TagDep**: the joint POS tagging and dependency parsing model (Hatori et al., 2011), where the look-ahead features are omitted.[5]
- **SegTag+Dep/SegTag+Dep'**: a pipeline combination of SegTag and Dep or Dep'.
- **SegTag+TagDep**: a pipeline combination of SegTag and TagDep, where only the segmentation output of SegTag is used as input to TagDep; the output tags of TagDep are used for evaluation.
- **SegTagDep**: the proposed full joint model.

All of the models described above except Dep' are based on the same feature sets for segmentation and tagging (Zhang and Clark, 2008; Zhang and Clark, 2010) and dependency parsing (Huang and Sagae, 2010). Therefore, we can investigate the contribution of the joint approach through comparison with the pipeline and joint models.

[3]http://www.keenage.com/html/e_index.html

[4]http://zh.wikipedia.org/wiki

[5]We used the original implementation used in Hatori et al. (2011). In Hatori et al. (2011), we confirmed that omission of the look-ahead features results in a 0.26% decrease in the parsing accuracy on CTB-5d (dev).

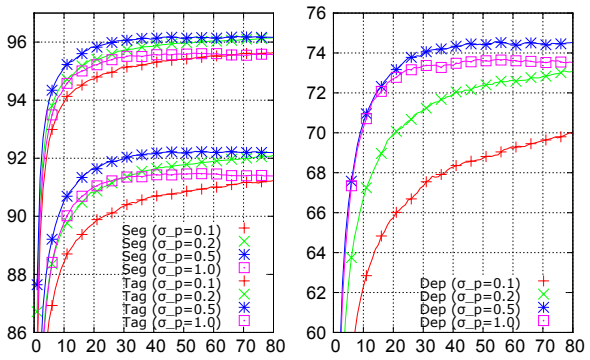

Figure 2: F1 scores (in %) of SegTagDep on CTB-5c-1 w.r.t. the training epoch (x-axis) and parsing feature weights (in legend).

4.3 Development Results

We have some parameters to tune: parsing feature weight σ_p, beam size, and training epoch. All these parameters are set based on experiments on CTB-5c. For experiments on CTB-5j, CTB-6, and CTB-7, the training epoch is set using the development set.

Figure 2 shows the F1 scores of the proposed model (SegTagDep) on CTB-5c-1 with respect to the training epoch and different parsing feature weights, where "Seg", "Tag", and "Dep" respectively denote the F1 scores of word segmentation, POS tagging, and dependency parsing. In this experiment, the external dictionaries are not used, and the beam size of 32 is used. Interestingly, if we simply set σ_p to 1, the accuracies seem to converge at lower levels. The $\sigma_p = 0.2$ setting seems to reach almost identical segmentation and tagging accuracies as the best setting $\sigma_p = 0.5$, but the convergence occurs more slowly. Based on this experiment, we set σ_p to 0.5 throughout the experiments in this paper.

Table 3 shows the performance and speed of the full joint model (with no dictionaries) on CTB-5c-1 with respect to the beam size. Although even the beam size of 32 results in competitive accuracies for word segmentation and POS tagging, the dependency accuracy is affected most by the increase of the beam size. Based on this experiment, we set the beam size of SegTagDep to 64 throughout the exper-

Beam	Seg	Tag	Dep	Speed
4	94.96	90.19	70.29	5.7
8	95.78	91.53	72.81	3.2
16	96.09	92.09	74.20	1.8
32	96.18	92.24	74.57	0.95
64	96.28	92.37	74.96	0.48

Table 3: F1 scores and speed (in sentences per sec.) of SegTagDep on CTB-5c-1 w.r.t. the beam size.

iments in this paper, unless otherwise noted.

4.4 Main Results

In this section, we present experimentally obtained results using the proposed and baseline models. Table 4 shows the segmentation, POS tagging, and dependency parsing F1 scores of these models on CTB-5c. Irrespective of the existence of the dictionary features, the joint model SegTagDep largely increases the POS tagging and dependency parsing accuracies (by 0.56–0.63% and 2.34–2.44%); the improvements in parsing accuracies are still significant even compared with SegTag+Dep' (the pipeline model with the look-ahead features). However, when the external dictionaries are not used ("wo/dict"), no substantial improvements for segmentation accuracies were observed. In contrast, when the dictionaries are used ("w/dict"), the segmentation accuracies are now improved over the baseline model SegTag consistently (on every trial). Although the overall improvement in segmentation is only around 0.1%, more than 1% improvement is observed if we specifically examine OOV[6] words. The difference between "wo/dict" and "w/dict" results suggests that the syntactic dependencies might work as a noise when the segmentation model is insufficiently stable, but the model does improve when it is stable, not receiving negative effects from the syntactic dependencies.

The partially joint model SegTag+TagDep is shown to perform reasonably well in dependency parsing: with dictionaries, it achieved the 2.02% improvement over SegTag+Dep, which is only 0.32% lower than SegTagDep. However, whereas SegTag+TagDep showed no substantial improvement in tagging accuracies over SegTag (when the dictionaries are used), SegTagDep achieved consistent improvements of 0.46% and 0.58% (without/with dictionaries); these differences can be attributed to the combination of the relieved error propagation and the incorporation of the syntactic dependencies. In addition, SegTag+TagDep has OOV tagging accuracies consistently lower than SegTag, suggesting that the syntactic dependency has a negative effect on the POS tagging accuracy of OOV words[7]. In contrast, this negative effect is not observed for SegTagDep: both the overall tagging accuracy and the OOV accuracy are improved, demonstrating the effectiveness of the proposed model.

Figure 3 shows the performance and processing time comparison of various models and their combinations. Although SegTagDep takes a few times longer to achieve accuracies comparable to those of SegTag+Dep/TagDep, it seems to present potential

[6]We define the OOV words as the words that have not seen in the training data, even when the external dictionaries are used.

System	Seg	Tag
Kruengkrai '09	97.87	93.67
Zhang '10	97.78	93.67
Sun '11	98.17	94.02
Wang '11	98.11	94.18
SegTag	97.66	93.61
SegTagDep	97.73	94.46
SegTag(d)	98.18	94.08
SegTagDep(d)	**98.26**	**94.64**

Table 5: Final results on CTB-5j

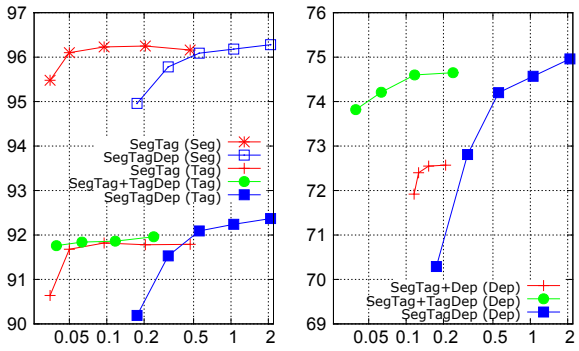

Figure 3: Performance of baseline and joint models w.r.t. the average processing time (in sec.) per sentence. Each point corresponds to the beam size of 4, 8, 16, 32, (64). The beam size of 16 is used for SegTag in SegTag+Dep and SegTag+TagDep.

[7]This is consistent with Hatori et al. (2011)'s observation that although the joint POS tagging and dependency parsing improves the accuracy of syntactically influential POS tags, it has a slight side effect of increasing the confusion between general and proper nouns (NN vs. NR).

	Model	Segmentation ALL	Segmentation OOV	POS Tagging ALL	POS Tagging OOV	Dependency
wo/dict	SegTag+Dep	**96.22**	72.24	91.74	59.82	72.58
	SegTag+Dep'					72.94 (+0.36‡)
	SegTag+TagDep			91.86 (+0.12‡)	58.89 (-0.93‡)	74.60 (+2.02‡)
	SegTagDep	96.19 (-0.03)	72.24 (+0.00)	**92.30** (+0.56‡)	**61.03** (+1.21‡)	**74.92** (+2.34‡)
w/dict	SegTag+Dep	96.82	78.32	92.34	65.44	73.53
	SegTag+Dep'					73.90 (+0.37‡)
	SegTag+TagDep			92.35 (+0.01)	63.20 (-2.24‡)	75.45 (+1.92‡)
	SegTagDep	**96.90** (+0.08‡)	**79.38** (+1.06‡)	**92.97** (+0.63‡)	**67.40** (+1.96‡)	**75.97** (+2.44‡)

Table 4: Segmentation, POS tagging, and (unlabeled attachment) dependency F1 scores averaged over five trials on CTB-5c. Figures in parentheses show the differences over SegTag+Dep (‡ : $p < 0.01$).

for greater improvement, especially for tagging and parsing accuracies, when a larger beam can be used.

4.5 Comparison with Other Systems

Table 5 and Table 6 show a comparison of the segmentation and POS tagging accuracies with other state-of-the-art models. "Kruengkrai+ '09" is a lattice-based model by Kruengkrai et al. (2009). "Zhang '10" is the incremental model by Zhang and Clark (2010). These two systems use no external resources other than the CTB corpora. "Sun+ '11" is a CRF-based model (Sun, 2011) that uses a combination of several models, with a dictionary of idioms. "Wang+ '11" is a semi-supervised model by Wang et al. (2011), which additionally uses the Chinese Gigaword Corpus.

Our models with dictionaries (those marked with '(d)') have competitive accuracies to other state-of-the-art systems, and SegTagDep(d) achieved the best reported segmentation and POS tagging accuracies, using no additional corpora other than the dictionaries. Particularly, the POS tagging accuracy is more than 0.4% higher than the previous best system thanks to the contribution of syntactic dependencies. These results also suggest that the use of readily available dictionaries can be more effective than semi-supervised approaches.

Model	CTB-6 Test Seg	Tag	Dep	CTB-7 Test Seg	Tag	Dep
Kruengkrai '09	95.50	90.50	-	95.40	89.86	-
Wang '11	95.79	91.12	-	95.65	90.46	-
SegTag+Dep	95.46	90.64	72.57	95.49	90.11	71.25
SegTagDep	95.45	91.27	74.88	95.42	90.62	73.58
(diff.)	-0.01	+0.63‡	+2.31‡	-0.07	+0.51‡	+2.33‡
SegTag+Dep(d)	96.13	91.38	73.62	95.98	90.68	72.06
SegTagDep(d)	**96.18**	**91.95**	**75.76**	**96.07**	**91.28**	**74.58**
(diff.)	+0.05	+0.57‡	+2.14‡	+0.09‡	+0.60‡	+2.52‡

Table 6: Final results on CTB-6 and CTB-7

5 Conclusion

In this paper, we proposed the first joint model for word segmentation, POS tagging, and dependency parsing in Chinese. The model demonstrated substantial improvements on the three tasks over the pipeline combination of the state-of-the-art joint segmentation and POS tagging model, and dependency parser. Particularly, results showed that the accuracies of POS tagging and dependency parsing were remarkably improved by 0.6% and 2.4%, respectively corresponding to 8.3% and 10.2% error reduction. For word segmentation, although the overall improvement was only around 0.1%, greater than 1% improvements was observed for OOV words. We conducted some comparison experiments of the partially joint and full joint models. Compared to SegTagDep, SegTag+TagDep performs reasonably well in terms of dependency parsing accuracy, whereas the POS tagging accuracies are more than 0.5% lower.

In future work, probabilistic pruning techniques such as the one based on a maximum entropy model are expected to improve the efficiency of the joint model further because the accuracies are apparently still improved if a larger beam can be used. More efficient decoding would also allow the use of the look-ahead features (Hatori et al., 2011) and richer parsing features (Zhang and Nivre, 2011).

Acknowledgement We are grateful to the anonymous reviewers for their comments and suggestions, and to Xianchao Wu, Kun Yu, Pontus Stenetorp, and Shinsuke Mori for their helpful feedback.

References

Shay B. Cohen and Noah A. Smith. 2007. Joint morphological and syntactic disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*.

Michael Collins and Brian Roark. 2004. Incremental parsing with the perceptron algorithm. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL 2004)*.

Xiangyu Duan, Jun Zhao, and Bo Xu. 2007. Probabilistic parsing action models for multi-lingual dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*.

Yoav Goldberg and Reut Tsarfaty. 2008. A single generative model for joint morphological segmentation and syntactic parsing. In *Proceedings of the 46th Annual Meeting of the Association of Computational Linguistics (ACL-2008)*.

Jun Hatori, Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. 2011. Incremental joint POS tagging and dependency parsing in Chinese. In *Proceedings of the Fifth International Joint Conference on Natural Language Processing (IJCNLP-2011)*.

Liang Huang and Kenji Sagae. 2010. Dynamic programming for linear-time incremental parsing. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*.

Wenbin Jiang, Liang Huang, Qun Liu, and Yajuan Lu. 2008. A cascaded linear model for joint Chinese word segmentation and part-of-speech tagging. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.

Canasai Kruengkrai, Kiyotaka Uchimoto, Jun'ichi Kazama, Yiou Wang, Kentaro Torisawa, and Hitoshi Isahara. 2009. An error-driven word-character hybrid model for joint Chinese word segmentation and POS tagging. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*.

Zhenghua Li, Min Zhang, Wanxiang Che, Ting Liu, Wenliang Chen, and Haizhou Haizhou. 2011. Joint models for Chinese POS tagging and dependency parsing. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*.

Xiaoqiang Luo. 2003. A maximum entropy Chinese character-based parser. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP-2003)*.

Richard Sproat, Chilin Shih, William Gale, and Nancy Chang. 1996. A stochastic finite-state word-segmentation algorithm for Chinese. *Computational Linguistics*, 22.

Weiwei Sun. 2011. A stacked sub-word model for joint Chinese word segmentation and part-of-speech tagging. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.

Yiou Wang, Jun'ichi Kazama, Yoshimasa Tsuruoka, Wenliang Chen, Yujie Zhang, and Kentaro Torisawa. 2011. Improving Chinese word segmentation and POS tagging with semi-supervised methods using large auto-analyzed data. In *Proceedings of the Fifth International Joint Conference on Natural Language Processing (IJCNLP-2011)*.

Fei Xia. 2000. The part-of-speech tagging guidelines for the Penn Chinese treebank (3.0). Technical Report IRCS-00-07, University of Pennsylvania Institute for Research in Cognitive Science Technical Report, October.

Yue Zhang and Stephen Clark. 2008. Joint word segmentation and POS tagging using a single perceptron. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.

Yue Zhang and Stephen Clark. 2010. A fast decoder for joint word segmentation and POS-tagging using a single discriminative model. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*.

Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (short papers)*.