Joint Word Segmentation and POS Tagging using a Single Perceptron

Yue Zhang and **Stephen Clark**
Oxford University Computing Laboratory
Wolfson Building, Parks Road
Oxford OX1 3QD, UK
{yue.zhang,stephen.clark}@comlab.ox.ac.uk

Abstract

For Chinese POS tagging, word segmentation is a preliminary step. To avoid error propagation and improve segmentation by utilizing POS information, segmentation and tagging can be performed simultaneously. A challenge for this joint approach is the large combined search space, which makes efficient decoding very hard. Recent research has explored the integration of segmentation and POS tagging, by decoding under restricted versions of the full combined search space. In this paper, we propose a joint segmentation and POS tagging model that does not impose any hard constraints on the interaction between word and POS information. Fast decoding is achieved by using a novel multiple-beam search algorithm. The system uses a discriminative statistical model, trained using the generalized perceptron algorithm. The joint model gives an error reduction in segmentation accuracy of 14.6% and an error reduction in tagging accuracy of 12.2%, compared to the traditional pipeline approach.

1 Introduction

Since Chinese sentences do not contain explicitly marked word boundaries, word segmentation is a necessary step before POS tagging can be performed. Typically, a Chinese POS tagger takes segmented inputs, which are produced by a separate word segmentor. This two-step approach, however, has an obvious flaw of error propagation, since word segmentation errors cannot be corrected by the POS tagger. A better approach would be to utilize POS information to improve word segmentation. For example, the POS-word pattern "number word" + "个 (a common measure word)" can help in segmenting the character sequence "一个人" into the word sequence "一 (one) 个 (measure word) 人 (person)" instead of "一 (one) 个人 (personal; adj)". Moreover, the comparatively rare POS pattern "number word" + "number word" can help to prevent segmenting a long number word into two words.

In order to avoid error propagation and make use of POS information for word segmentation, segmentation and POS tagging can be viewed as a single task: given a raw Chinese input sentence, the joint POS tagger considers all possible segmented and tagged sequences, and chooses the overall best output. A major challenge for such a joint system is the large search space faced by the decoder. For a sentence with n characters, the number of possible output sequences is $O(2^{n-1} \cdot T^n)$, where T is the size of the tag set. Due to the nature of the combined candidate items, decoding can be inefficient even with dynamic programming.

Recent research on Chinese POS tagging has started to investigate joint segmentation and tagging, reporting accuracy improvements over the pipeline approach. Various decoding approaches have been used to reduce the combined search space. Ng and Low (2004) mapped the joint segmentation and POS tagging task into a single character sequence tagging problem. Two types of tags are assigned to each character to represent its segmentation and POS. For example, the tag "b_NN" indicates a character at the beginning of a noun. Using this method, POS features are allowed to interact with segmentation.

Proceedings of ACL-08: HLT, pages 888–896,
Columbus, Ohio, USA, June 2008.

Since tagging is restricted to characters, the search space is reduced to $O((4T)^n)$, and beam search decoding is effective with a small beam size. However, the disadvantage of this model is the difficulty in incorporating whole word information into POS tagging. For example, the standard "word + POS tag" feature is not explicitly applicable. Shi and Wang (2007) introduced POS information to segmentation by reranking. N-best segmentation outputs are passed to a separately-trained POS tagger, and the best output is selected using the overall POS-segmentation probability score. In this system, the decoding for word segmentation and POS tagging are still performed separately, and exact inference for both is possible. However, the interaction between POS and segmentation is restricted by reranking: POS information is used to improve segmentation only for the N segmentor outputs.

In this paper, we propose a novel joint model for Chinese word segmentation and POS tagging, which does not limiting the interaction between segmentation and POS information in reducing the combined search space. Instead, a novel multiple beam search algorithm is used to do decoding efficiently. Candidate ranking is based on a discriminative joint model, with features being extracted from segmented words and POS tags simultaneously. The training is performed by a single generalized perceptron (Collins, 2002). In experiments with the Chinese Treebank data, the joint model gave an error reduction of 14.6% in segmentation accuracy and 12.2% in the overall segmentation and tagging accuracy, compared to the traditional pipeline approach. In addition, the overall results are comparable to the best systems in the literature, which exploit knowledge outside the training data, even though our system is fully data-driven.

Different methods have been proposed to reduce error propagation between pipelined tasks, both in general (Sutton et al., 2004; Daumé III and Marcu, 2005; Finkel et al., 2006) and for specific problems such as language modeling and utterance classification (Saraclar and Roark, 2005) and labeling and chunking (Shimizu and Haas, 2006). Though our model is built specifically for Chinese word segmentation and POS tagging, the idea of using the perceptron model to solve multiple tasks simultaneously can be generalized to other tasks.

1	word w
2	word bigram w_1w_2
3	single-character word w
4	a word of length l with starting character c
5	a word of length l with ending character c
6	space-separated characters c_1 and c_2
7	character bigram c_1c_2 in any word
8	the first / last characters c_1 / c_2 of any word
9	word w immediately before character c
10	character c immediately before word w
11	the starting characters c_1 and c_2 of two consecutive words
12	the ending characters c_1 and c_2 of two consecutive words
13	a word of length l with previous word w
14	a word of length l with next word w

Table 1: Feature templates for the baseline segmentor

2 The Baseline System

We built a two-stage baseline system, using the perceptron segmentation model from our previous work (Zhang and Clark, 2007) and the perceptron POS tagging model from Collins (2002). We use *baseline system* to refer to the system which performs segmentation first, followed by POS tagging (using the single-best segmentation); *baseline segmentor* to refer to the segmentor from (Zhang and Clark, 2007) which performs segmentation only; and *baseline POStagger* to refer to the Collins tagger which performs POS tagging only (given segmentation). The features used by the baseline segmentor are shown in Table 1. The features used by the POS tagger, some of which are different to those from Collins (2002) and are specific to Chinese, are shown in Table 2.

The word segmentation features are extracted from word bigrams, capturing word, word length and character information in the context. The word length features are normalized, with those more than 15 being treated as 15.

The POS tagging features are based on contextual information from the tag trigram, as well as the neighboring three-word window. To reduce overfitting and increase the decoding speed, templates 4, 5, 6 and 7 only include words with less than 3 characters. Like the baseline segmentor, the baseline tagger also normalizes word length features.

1	tag t with word w
2	tag bigram t_1t_2
3	tag trigram $t_1t_2t_3$
4	tag t followed by word w
5	word w followed by tag t
6	word w with tag t and previous character c
7	word w with tag t and next character c
8	tag t on single-character word w in character trigram c_1wc_2
9	tag t on a word starting with char c
10	tag t on a word ending with char c
11	tag t on a word containing char c (not the starting or ending character)
12	tag t on a word starting with char c_0 and containing char c
13	tag t on a word ending with char c_0 and containing char c
14	tag t on a word containing repeated char cc
15	tag t on a word starting with character category g
16	tag t on a word ending with character category g

Table 2: Feature templates for the baseline POS tagger

Templates 15 and 16 in Table 2 are inspired by the CTBMorph feature templates in Tseng et al. (2005), which gave the most accuracy improvement in their experiments. Here the category of a character is the set of tags seen on the character during training. Other morphological features from Tseng et al. (2005) are not used because they require extra web corpora besides the training data.

During training, the baseline POS tagger stores special word-tag pairs into a *tag dictionary* (Ratnaparkhi, 1996). Such information is used by the decoder to prune unlikely tags. For each word occurring more than N times in the training data, the decoder can only assign a tag the word has been seen with in the training data. This method led to improvement in the decoding speed as well as the output accuracy for English POS tagging (Ratnaparkhi, 1996). Besides tags for frequent words, our baseline POS tagger also uses the tag dictionary to store closed-set tags (Xia, 2000) – those associated only with a limited number of Chinese words.

3 Joint Segmentation and Tagging Model

In this section, we build a joint word segmentation and POS tagging model that uses exactly the same source of information as the baseline system, by applying the feature templates from the baseline word segmentor and POS tagger. No extra knowledge is used by the joint model. However, because word segmentation and POS tagging are performed simultaneously, POS information participates in word segmentation.

3.1 Formulation of the joint model

We formulate joint word segmentation and POS tagging as a single problem, which maps a raw Chinese sentence to a segmented and POS tagged output. Given an input sentence x, the output $F(x)$ satisfies:

$$F(x) = \arg\max_{y \in \text{GEN}(x)} \text{Score}(y)$$

where $\text{GEN}(x)$ represents the set of possible outputs for x.

$\text{Score}(y)$ is computed by a feature-based linear model. Denoting the global feature vector for the tagged sentence y with $\Phi(y)$, we have:

$$\text{Score}(y) = \Phi(y) \cdot \vec{w}$$

where $\vec{w}$ is the parameter vector in the model. Each element in $\vec{w}$ gives a weight to its corresponding element in $\Phi(y)$, which is the count of a particular feature over the whole sentence y. We calculate the $\vec{w}$ value by supervised learning, using the averaged perceptron algorithm (Collins, 2002), given in Figure 1. [1]

We take the union of feature templates from the baseline segmentor (Table 1) and POS tagger (Table 2) as the feature templates for the joint system. All features are treated equally and processed together according to the linear model, regardless of whether they are from the baseline segmentor or tagger. In fact, most features from the baseline POS tagger, when used in the joint model, represent segmentation patterns as well. For example, the aforementioned pattern "number word" + "个", which is

[1] In order to provide a comparison for the perceptron algorithm we also tried SVM$^{\text{struct}}$ (Tsochantaridis et al., 2004) for parameter estimation, but this training method was prohibitively slow.

Inputs: training examples (x_i, y_i)
Initialization: set $\vec{w} = 0$
Algorithm:
for $t = 1..T$, $i = 1..N$
calculate $z_i = \arg\max_{y \in \text{GEN}(x_i)} \Phi(y) \cdot \vec{w}$
if $z_i \neq y_i$
$\vec{w} = \vec{w} + \Phi(y_i) - \Phi(z_i)$
Outputs: $\vec{w}$

Figure 1: The perceptron learning algorithm

useful only for the POS "number word" in the baseline tagger, is also an effective indicator of the segmentation of the two words (especially "个") in the joint model.

3.2 The decoding algorithm

One of the main challenges for the joint segmentation and POS tagging system is the decoding algorithm. The speed and accuracy of the decoder is important for the perceptron learning algorithm, but the system faces a very large search space of combined candidates. Given the linear model and feature templates, exact inference is very hard even with dynamic programming.

Experiments with the standard beam-search decoder described in (Zhang and Clark, 2007) resulted in low accuracy. This beam search algorithm processes an input sentence incrementally. At each stage, the incoming character is combined with existing partial candidates in all possible ways to generate new partial candidates. An agenda is used to control the search space, keeping only the B best partial candidates ending with the current character. The algorithm is simple and efficient, with a linear time complexity of $O(BTn)$, where n is the size of input sentence, and T is the size of the tag set ($T = 1$ for pure word segmentation). It worked well for word segmentation alone (Zhang and Clark, 2007), even with an agenda size as small as 8, and a simple beam search algorithm also works well for POS tagging (Ratnaparkhi, 1996). However, when applied to the joint model, it resulted in a reduction in segmentation accuracy (compared to the baseline segmentor) even with B as large as 1024.

One possible cause of the poor performance of the standard beam search method is the combined nature of the candidates in the search space. In the baseline POS tagger, candidates in the beam are tagged sequences ending with the current word, which can be compared directly with each other. However, for the joint problem, candidates in the beam are segmented and tagged sequences up to the current character, where the last word can be a complete word or a partial word. A problem arises in whether to give POS tags to incomplete words. If partial words are given POS tags, it is likely that some partial words are "justified" as complete words by the current POS information. On the other hand, if partial words are not given POS tag features, the correct segmentation for long words can be lost during partial candidate comparison (since many short completed words with POS tags are likely to be preferred to a long incomplete word with no POS tag features).[2]

Input: raw sentence $sent$ – a list of characters
Variables: candidate sentence $item$ – a list of (word, tag) pairs;
maximum word-length record $maxlen$ for each tag;
the agenda list $agendas$;
the tag dictionary $tagdict$;
$start\_index$ for current word;
end_index for current word
Initialization: $agendas[0] = [\text{“”}]$,
$agendas[i] = []$ $(i! = 0)$
Algorithm:
for $end\_index = 1$ to $sent$.length:
foreach tag:
for $start\_index =$
max$(1,\ end_index - maxlen[tag] + 1)$
to end_index:
$word = sent[start\_index..end\_index]$
if $(word, tag)$ consistent with $tagdict$:
for $item \in agendas[start\_index - 1]$:
$item_1 = item$
$item_1$.append(($word$,tag))
$agendas[end\_index]$.insert($item_1$)
Outputs: $agendas[sent$.length].best_item

Figure 2: The decoding algorithm for the joint word segmentor and POS tagger

[2] We experimented with both assigning POS features to partial words and omitting them; the latter method performed better but both performed significantly worse than the multiple beam search method described below.

Another possible cause is the exponential growth in the number of possible candidates with increasing sentence size. The number increases from $O(T^n)$ for the baseline POS tagger to $O(2^{n-1}T^n)$ for the joint system. As a result, for an incremental decoding algorithm, the number of possible candidates increases exponentially with the current word or character index. In the POS tagging problem, a new incoming word enlarges the number of possible candidates by a factor of T (the size of the tag set). For the joint problem, however, the enlarging factor becomes $2T$ with each incoming character. The speed of search space expansion is much faster, but the number of candidates is still controlled by a single, fixed-size beam at any stage. If we assume that the beam is not large enough for all the candidates at at each stage, then, from the newly generated candidates, the baseline POS tagger can keep $1/T$ for the next processing stage, while the joint model can keep only $1/2T$, and has to discard the rest. Therefore, even when the candidate comparison standard is ignored, we can still see that the chance for the overall best candidate to fall out of the beam is largely increased. Since the search space growth is exponential, increasing the fixed beam size is not effective in solving the problem.

To solve the above problems, we developed a multiple beam search algorithm, which compares candidates only with complete tagged words, and enables the size of the search space to scale with the input size. The algorithm is shown in Figure 2. In this decoder, an agenda is assigned to each character in the input sentence, recording the B best segmented and tagged partial candidates ending with the character. The input sentence is still processed incrementally. However, now when a character is processed, existing partial candidates ending with any previous characters are available. Therefore, the decoder enumerates all possible tagged words ending with the current character, and combines each word with the partial candidates ending with its previous character. All input characters are processed in the same way, and the final output is the best candidate in the final agenda. The time complexity of the algorithm is $O(WTBn)$, with W being the maximum word size, T being the total number of POS tags and n the number of characters in the input. It is also linear in the input size. Moreover, the decoding algorithm gives competent accuracy with a small agenda size of $B = 16$.

To further limit the search space, two optimizations are used. First, the maximum word length for each tag is recorded and used by the decoder to prune unlikely candidates. Because the majority of tags only apply to words with length 1 or 2, this method has a strong effect. Development tests showed that it improves the speed significantly, while having a very small negative influence on the accuracy. Second, like the baseline POS tagger, the tag dictionary is used for Chinese closed set tags and the tags for frequent words. To words outside the tag dictionary, the decoder still tries to assign every possible tag.

3.3 Online learning

Apart from features, the decoder maintains other types of information, including the tag dictionary, the word frequency counts used when building the tag dictionary, the maximum word lengths by tag, and the character categories. The above data can be collected by scanning the corpus before training starts. However, in both the baseline tagger and the joint POS tagger, they are updated incrementally during the perceptron training process, consistent with online learning.[3]

The online updating of word frequencies, maximum word lengths and character categories is straightforward. For the online updating of the tag dictionary, however, the decision for frequent words must be made dynamically because the word frequencies keep changing. This is done by caching the number of occurrences of the current most frequent word M, and taking all words currently above the threshold $M/5000 + 5$ as frequent words. 5000 is a rough figure to control the number of frequent words, set according to Zipf's law. The parameter 5 is used to force all tags to be enumerated before a word is seen more than 5 times.

4 Related Work

Ng and Low (2004) and Shi and Wang (2007) were described in the Introduction. Both models reduced

[3]We took this approach because we wanted the whole training process to be online. However, for comparison purposes, we also tried precomputing the above information before training and the difference in performance was negligible.

the large search space by imposing strong restrictions on the form of search candidates. In particular, Ng and Low (2004) used character-based POS tagging, which prevents some important POS tagging features such as word + POS tag; Shi and Wang (2007) used an N-best reranking approach, which limits the influence of POS tagging on segmentation to the N-best list. In comparison, our joint model does not impose any hard limitations on the interaction between segmentation and POS information.[4] Fast decoding speed is achieved by using a novel multiple-beam search algorithm.

Nakagawa and Uchimoto (2007) proposed a hybrid model for word segmentation and POS tagging using an HMM-based approach. Word information is used to process known-words, and character information is used for unknown words in a similar way to Ng and Low (2004). In comparison, our model handles character and word information simultaneously in a single perceptron model.

5 Experiments

The Chinese Treebank (CTB) 4 is used for the experiments. It is separated into two parts: CTB 3 (420K characters in 150K words / 10364 sentences) is used for the final 10-fold cross validation, and the rest (240K characters in 150K words / 4798 sentences) is used as training and test data for development.

The standard F-scores are used to measure both the word segmentation accuracy and the overall segmentation and tagging accuracy, where the overall accuracy is $TF = 2pr/(p + r)$, with the precision p being the percentage of correctly segmented and tagged words in the decoder output, and the recall r being the percentage of gold-standard tagged words that are correctly identified by the decoder. For direct comparison with Ng and Low (2004), the POS tagging accuracy is also calculated by the percentage of correct tags on each character.

5.1 Development experiments

The learning curves of the baseline and joint models are shown in Figure 3, Figure 4 and Figure 5, respectively. These curves are used to show the convergence of perceptron and decide the number of training iterations for the test. It should be noticed that the accuracies from Figure 4 and Figure 5 are not comparable because gold-standard segmentation is used as the input for the baseline tagger. According to the figures, the number of training iterations

[4] Apart from the beam search algorithm, we do impose some minor limitations on the search space by methods such as the tag dictionary, but these can be seen as optional pruning methods for optimization.

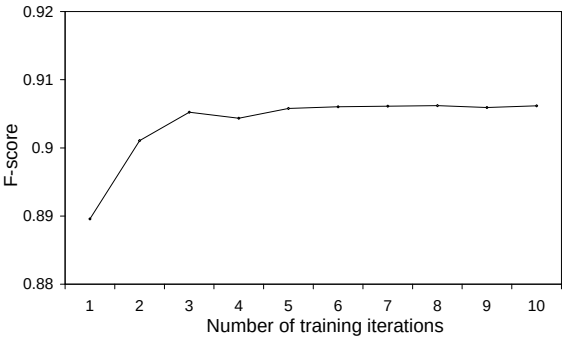

Figure 3: The learning curve of the baseline segmentor

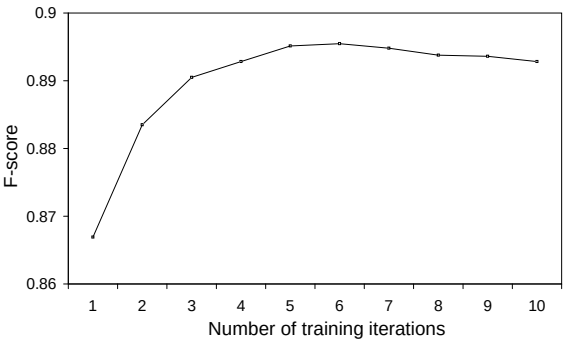

Figure 4: The learning curve of the baseline tagger

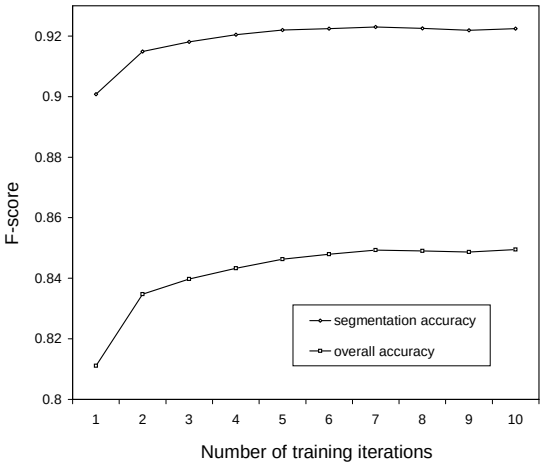

Figure 5: The learning curves of the joint system

Tag	Seg	NN	NR	VV	AD	JJ	CD
NN	20.47	–	0.78	4.80	0.67	2.49	0.04
NR	5.95	3.61	–	0.19	0.04	0.07	0
VV	12.13	6.51	0.11	–	0.93	0.56	0.04
AD	3.24	0.30	0	0.71	–	0.33	0.22
JJ	3.09	0.93	0.15	0.26	0.26	–	0.04
CD	1.08	0.04	0	0	0.07	0	–

Table 3: Error analysis for the joint model

for the baseline segmentor, POS tagger, and the joint system are set to 8, 6, and 7, respectively for the remaining experiments.

There are many factors which can influence the accuracy of the joint model. Here we consider the special character category features and the effect of the tag dictionary. The character category features (templates 15 and 16 in Table 2) represent a Chinese character by all the tags associated with the character in the training data. They have been shown to improve the accuracy of a Chinese POS tagger (Tseng et al., 2005). In the joint model, these features also represent segmentation information, since they concern the starting and ending characters of a word. Development tests showed that the overall tagging F-score of the joint model increased from 84.54% to 84.93% using the character category features. In the development test, the use of the tag dictionary improves the decoding speed of the joint model, reducing the decoding time from 416 seconds to 256 seconds. The overall tagging accuracy also increased slightly, consistent with observations from the pure POS tagger.

The error analysis for the development test is shown in Table 3. Here an error is counted when a word in the standard output is not produced by the decoder, due to incorrect segmentation or tag assignment. Statistics about the six most frequently mistaken tags are shown in the table, where each row presents the analysis of one tag from the standard output, and each column gives a wrongly assigned value. The column "Seg" represents segmentation errors. Each figure in the table shows the percentage of the corresponding error from all the errors.

It can be seen from the table that the NN-VV and VV-NN mistakes were the most commonly made by the decoder, while the NR-NN mistakes are also frequent. These three types of errors significantly outnumber the rest, together contributing 14.92% of all the errors. Moreover, the most commonly mistaken tags are NN and VV, while among the most frequent tags in the corpus, PU, DEG and M had comparatively less errors. Lastly, segmentation errors contribute around half (51.47%) of all the errors.

	Baseline			Joint		
#	SF	TF	TA	SF	TF	TA
1	96.98	92.91	94.14	97.21	93.46	94.66
2	97.16	93.20	94.34	97.62	93.85	94.79
3	95.02	89.53	91.28	95.94	90.86	92.38
4	95.51	90.84	92.55	95.92	91.60	93.31
5	95.49	90.91	92.57	96.06	91.72	93.25
6	93.50	87.33	89.87	94.56	88.83	91.14
7	94.48	89.44	91.61	95.30	90.51	92.41
8	93.58	88.41	90.93	95.12	90.30	92.32
9	93.92	89.15	91.35	94.79	90.33	92.45
10	96.31	91.58	93.01	96.45	91.96	93.45
Av.	95.20	90.33	92.17	95.90	91.34	93.02

Table 4: The accuracies by 10-fold cross validation

SF – segmentation F-score,
TF – overall F-score,
TA – tagging accuracy by character.

5.2 Test results

10-fold cross validation is performed to test the accuracy of the joint word segmentor and POS tagger, and to make comparisons with existing models in the literature. Following Ng and Low (2004), we partition the sentences in CTB 3, ordered by sentence ID, into 10 groups evenly. In the nth test, the nth group is used as the testing data.

Table 4 shows the detailed results for the cross validation tests, each row representing one test. As can be seen from the table, the joint model outperforms the baseline system in each test.

Table 5 shows the overall accuracies of the baseline and joint systems, and compares them to the relevant models in the literature. The accuracy of each model is shown in a row, where "Ng" represents the models from Ng and Low (2004) and "Shi" represents the models from Shi and Wang (2007). Each accuracy measure is shown in a column, including the segmentation F-score (SF), the overall tagging

Model	SF	TF	TA
Baseline+ (Ng)	95.1	–	91.7
Joint+ (Ng)	95.2	–	91.9
Baseline+* (Shi)	95.85	91.67	–
Joint+* (Shi)	96.05	91.86	–
Baseline (ours)	95.20	90.33	92.17
Joint (ours)	95.90	91.34	93.02

Table 5: The comparison of overall accuracies by 10-fold cross validation using CTB

+ – knowledge about sepcial characters,
* – knowledge from semantic net outside CTB.

F-score (TF) and the tagging accuracy by characters (TA). As can be seen from the table, our joint model achieved the largest improvement over the baseline, reducing the segmentation error by 14.58% and the overall tagging error by 12.18%.

The overall tagging accuracy of our joint model was comparable to but less than the joint model of Shi and Wang (2007). Despite the higher accuracy improvement from the baseline, the joint system did not give higher overall accuracy. One likely reason is that Shi and Wang (2007) included knowledge about special characters and semantic knowledge from web corpora (which may explain the higher baseline accuracy), while our system is completely data-driven. However, the comparison is indirect because our partitions of the CTB corpus are different. Shi and Wang (2007) also chunked the sentences before doing 10-fold cross validation, but used an uneven split. We chose to follow Ng and Low (2004) and split the sentences evenly to facilitate further comparison.

Compared with Ng and Low (2004), our baseline model gave slightly better accuracy, consistent with our previous observations about the word segmentors (Zhang and Clark, 2007). Due to the large accuracy gain from the baseline, our joint model performed much better.

In summary, when compared with existing joint word segmentation and POS tagging systems in the literature, our proposed model achieved the best accuracy boost from the cascaded baseline, and competent overall accuracy.

6 Conclusion and Future Work

We proposed a joint Chinese word segmentation and POS tagging model, which achieved a considerable reduction in error rate compared to a baseline two-stage system.

We used a single linear model for combined word segmentation and POS tagging, and chose the generalized perceptron algorithm for joint training. and beam search for efficient decoding. However, the application of beam search was far from trivial because of the size of the combined search space. Motivated by the question of what are the comparable partial hypotheses in the space, we developed a novel multiple beam search decoder which effectively explores the large search space. Similar techniques can potentially be applied to other problems involving joint inference in NLP.

Other choices are available for the decoding of a joint linear model, such as exact inference with dynamic programming, provided that the range of features allows efficient processing. The baseline feature templates for Chinese segmentation and POS tagging, when added together, makes exact inference for the proposed joint model very hard. However, the accuracy loss from the beam decoder, as well as alternative decoding algorithms, are worth further exploration.

The joint system takes features only from the baseline segmentor and the baseline POS tagger to allow a fair comparison. There may be additional features that are particularly useful to the joint system. Open features, such as knowledge of numbers and European letters, and relationships from semantic networks (Shi and Wang, 2007), have been reported to improve the accuracy of segmentation and POS tagging. Therefore, given the flexibility of the feature-based linear model, an obvious next step is the study of open features in the joint segmentor and POS tagger.

Acknowledgements

We thank Hwee-Tou Ng and Mengqiu Wang for their helpful discussions and sharing of experimental data, and the anonymous reviewers for their suggestions. This work is supported by the ORS and Clarendon Fund.

References

Michael Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the EMNLP conference*, pages 1–8, Philadelphia, PA.

Hal Daumé III and Daniel Marcu. 2005. Learning as search optimization: Approximate large margin methods for structured prediction. In *Proceedings of the ICML Conference*, pages 169–176, Bonn, Germany.

Jenny Rose Finkel, Christopher D. Manning, and Andrew Y. Ng. 2006. Solving the problem of cascading errors: Approximate Bayesian inference for linguistic annotation pipelines. In *Proceedings of the EMNLP Conference*, pages 618–626, Sydney, Australia.

Tetsuji Nakagawa and Kiyotaka Uchimoto. 2007. A hybrid approach to word segmentation and pos tagging. In *Proceedings of ACL Demo and Poster Session*, pages 217–220, Prague, Czech Republic.

Hwee Tou Ng and Jin Kiat Low. 2004. Chinese part-of-speech tagging: One-at-a-time or all-at-once? Word-based or character-based? In *Proceedings of the EMNLP Conference*, pages 277–284, Barcelona, Spain.

Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of the EMNLP Conference*, pages 133–142, Philadelphia, PA.

Murat Saraclar and Brian Roark. 2005. Joint discriminative language modeling and utterance classification. In *Proceedings of the ICASSP Conference*, volume 1, Philadelphia, USA.

Yanxin Shi and Mengqiu Wang. 2007. A dual-layer CRF based joint decoding method for cascade segmentation and labelling tasks. In *Proceedings of the IJCAI Conference*, Hyderabad, India.

Nobuyuki Shimizu and Andrew Haas. 2006. Exact decoding for jointly labeling and chunking sequences. In *Proceedings of the COLING/ACL Conference, Poster Sessions*, Sydney, Australia.

Charles Sutton, Khashayar Rohanimanesh, and Andrew McCallum. 2004. Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. In *Proceedings of the ICML Conference*, Banff, Canada.

Huihsin Tseng, Daniel Jurafsky, and Christopher Manning. 2005. Morphological features help POS tagging of unknown words across language varieties. In *Proceedings of the Fourth SIGHAN Workshop*, Jeju Island, Korea.

I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. 2004. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the ICML Conference*, Banff, Canada.

Fei Xia. 2000. The part-of-speech tagging guidelines for the Chinese Treebank (3.0). *IRCS Report, University of Pennsylvania*.

Yue Zhang and Stephen Clark. 2007. Chinese segmentation with a word-based perceptron algorithm. In *Proceedings of the ACL Conference*, pages 840–847, Prague, Czech Republic.