Automatic Adaptation of Annotation Standards: Chinese Word Segmentation and POS Tagging – A Case Study

Wenbin Jiang [†] **Liang Huang** [‡] **Qun Liu** [†]

[†]Key Lab. of Intelligent Information Processing
Institute of Computing Technology
Chinese Academy of Sciences
P.O. Box 2704, Beijing 100190, China
{jiangwenbin, liuqun}@ict.ac.cn

[‡]Google Research
1350 Charleston Rd.
Mountain View, CA 94043, USA
lianghuang@google.com
liang.huang.sh@gmail.com

Abstract

Manually annotated corpora are valuable but scarce resources, yet for many annotation tasks such as treebanking and sequence labeling there exist multiple corpora with *different* and *incompatible* annotation guidelines or standards. This seems to be a great waste of human efforts, and it would be nice to automatically adapt one annotation standard to another. We present a simple yet effective strategy that transfers knowledge from a differently annotated corpus to the corpus with desired annotation. We test the efficacy of this method in the context of Chinese word segmentation and part-of-speech tagging, where no segmentation and POS tagging standards are widely accepted due to the lack of morphology in Chinese. Experiments show that adaptation from the much larger People's Daily corpus to the smaller but more popular Penn Chinese Treebank results in significant improvements in both segmentation and tagging accuracies (with error reductions of 30.2% and 14%, respectively), which in turn helps improve Chinese parsing accuracy.

1 Introduction

Much of statistical NLP research relies on some sort of manually annotated corpora to train their models, but these resources are extremely expensive to build, especially at a large scale, for example in treebanking (Marcus et al., 1993). However the linguistic theories underlying these annotation efforts are often heavily debated, and as a result there often exist multiple corpora for the same task with vastly different and incompatible annotation philosophies. For example just for English treebanking there have been the Chomskian-style Penn Treebank (Marcus et al., 1993) the HPSG LinGo Redwoods Treebank (Oepen et al., 2002), and a smaller dependency treebank (Buchholz and Marsi, 2006). A second, related problem is that the raw texts are also drawn from different domains, which for the above example range from financial news (PTB/WSJ) to transcribed dialog (LinGo). These two problems seem be a great waste in human efforts, and it would be nice if one could automatically adapt from one annotation standard and/or domain to another in order to exploit much larger datasets for better training. The second problem, domain adaptation, is very well-studied, e.g. by Blitzer et al. (2006) and Daumé III (2007) (and see below for discussions), so in this paper we focus on the less studied, but equally important problem of *annotation-style adaptation.*

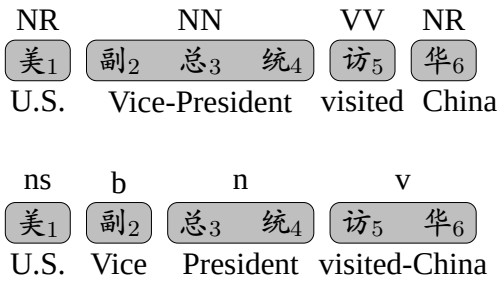

Figure 1: Incompatible word segmentation and POS tagging standards between CTB (upper) and People's Daily (below).

We present a very simple yet effective strategy that enables us to utilize knowledge from a differently annotated corpora for the training of a model on a corpus with desired annotation. The basic idea is very simple: we first train on a source corpus, resulting in a source classifier, which is used to label the target corpus and results in a "source-style" annotation of the target corpus. We then

train a second model on the target corpus with the first classifier's prediction as additional features for guided learning.

This method is very similar to some ideas in domain adaptation (Daumé III and Marcu, 2006; Daumé III, 2007), but we argue that the underlying problems are quite different. Domain adaptation assumes the labeling guidelines are preserved between the two domains, e.g., an adjective is always labeled as JJ regardless of from Wall Street Journal (WSJ) or Biomedical texts, and only the *distributions* are different, e.g., the word "control" is most likely a verb in WSJ but often a noun in Biomedical texts (as in "control experiment"). Annotation-style adaptation, however, tackles the problem where the guideline itself is changed, for example, one treebank might distinguish between transitive and intransitive verbs, while merging the different noun types (NN, NNS, etc.), and for example one treebank (PTB) might be much flatter than the other (LinGo), not to mention the fundamental disparities between their underlying linguistic representations (CFG vs. HPSG). In this sense, the problem we study in this paper seems much harder and more motivated from a linguistic (rather than statistical) point of view. More interestingly, our method, without any assumption on the distributions, can be simultaneously applied to both domain *and* annotation standards adaptation problems, which is very appealing in practice because the latter problem often implies the former, as in our case study.

To test the efficacy of our method we choose Chinese word segmentation and part-of-speech tagging, where the problem of incompatible annotation standards is one of the most evident: so far no segmentation standard is widely accepted due to the lack of a clear definition of Chinese *words*, and the (almost complete) lack of morphology results in much bigger ambiguities and heavy debates in tagging philosophies for Chinese parts-of-speech. The two corpora used in this study are the much larger People's Daily (PD) (5.86M words) corpus (Yu et al., 2001) and the smaller but more popular Penn Chinese Treebank (CTB) (0.47M words) (Xue et al., 2005). They used very different segmentation standards as well as different POS tagsets and tagging guidelines. For example, in Figure 1, People's Daily breaks "Vice-President" into two words while combines the phrase "visited-China" as a compound. Also CTB has four verbal categories (VV for normal verbs, and VC for copulas, etc.) while PD has only one verbal tag (v) (Xia, 2000). It is preferable to transfer knowledge from PD to CTB because the latter also annotates tree structures which is very useful for downstream applications like parsing, summarization, and machine translation, yet it is much smaller in size. Indeed, many recent efforts on Chinese-English translation and Chinese parsing use the CTB as the *de facto* segmentation and tagging standards, but suffers from the limited size of training data (Chiang, 2007; Bikel and Chiang, 2000). We believe this is also a reason why state-of-the-art accuracy for Chinese parsing is much lower than that of English (CTB is only half the size of PTB).

Our experiments show that adaptation from PD to CTB results in a significant improvement in segmentation and POS tagging, with error reductions of 30.2% and 14%, respectively. In addition, the improved accuracies from segmentation and tagging also lead to an improved parsing accuracy on CTB, reducing 38% of the error propagation from word segmentation to parsing. We envision this technique to be general and widely applicable to many other sequence labeling tasks.

In the rest of the paper we first briefly review the popular classification-based method for word segmentation and tagging (Section 2), and then describe our idea of annotation adaptation (Section 3). We then discuss other relevant previous work including co-training and classifier combination (Section 4) before presenting our experimental results (Section 5).

2 Segmentation and Tagging as Character Classification

Before describing the adaptation algorithm, we give a brief introduction of the baseline character classification strategy for segmentation, as well as joint segmenation and tagging (henceforth "Joint S&T"). following our previous work (Jiang et al., 2008). Given a Chinese sentence as sequence of n characters:

$$C_1\ C_2\ ..\ C_n$$

where C_i is a character, word segmentation aims to split the sequence into $m(\leq n)$ words:

$$C_{1:e_1}\ C_{e_1+1:e_2}\ ..\ C_{e_{m-1}+1:e_m}$$

where each subsequence $C_{i:j}$ indicates a Chinese word spanning from characters C_i to C_j (both in-

Algorithm 1 Perceptron training algorithm.

Input: Training examples (x_i, y_i)
$\vec{\alpha} \leftarrow \mathbf{0}$
for $t \leftarrow 1$ .. T **do**
  for $i \leftarrow 1$ .. N **do**
    $z_i \leftarrow \text{argmax}_{z \in \mathbf{GEN}(x_i)} \mathbf{\Phi}(x_i, z) \cdot \vec{\alpha}$
    if $z_i \neq y_i$ **then**
      $\vec{\alpha} \leftarrow \vec{\alpha} + \mathbf{\Phi}(x_i, y_i) - \mathbf{\Phi}(x_i, z_i)$
Output: Parameters $\vec{\alpha}$

clusive). While in Joint S&T, each word is further annotated with a POS tag:

$$C_{1:e_1}/t_1 \; C_{e_1+1:e_2}/t_2 \; .. \; C_{e_{m-1}+1:e_m}/t_m$$

where $t_k(k = 1..m)$ denotes the POS tag for the word $C_{e_{k-1}+1:e_k}$.

2.1 Character Classification Method

Xue and Shen (2003) describe for the first time the character classification approach for Chinese word segmentation, where each character is given a boundary tag denoting its relative position in a word. In Ng and Low (2004), Joint S&T can also be treated as a character classification problem, where a boundary tag is combined with a POS tag in order to give the POS information of the word containing these characters. In addition, Ng and Low (2004) find that, compared with POS tagging after word segmentation, Joint S&T can achieve higher accuracy on both segmentation and POS tagging. This paper adopts the tag representation of Ng and Low (2004). For word segmentation only, there are four boundary tags:

- *b*: the begin of the word
- *m*: the middle of the word
- *e*: the end of the word
- *s*: a single-character word

while for Joint S&T, a POS tag is attached to the tail of a boundary tag, to incorporate the word boundary information and POS information together. For example, *b-NN* indicates that the character is the begin of a noun. After all characters of a sentence are assigned boundary tags (or with POS postfix) by a classifier, the corresponding word sequence (or with POS) can be directly derived. Take segmentation for example, a character assigned a tag *s* or a subsequence of words assigned a tag sequence bm^*e indicates a word.

2.2 Training Algorithm and Features

Now we will show the training algorithm of the classifier and the features used. Several classification models can be adopted here, however, we choose the averaged perceptron algorithm (Collins, 2002) because of its simplicity and high accuracy. It is an online training algorithm and has been successfully used in many NLP tasks, such as POS tagging (Collins, 2002), parsing (Collins and Roark, 2004), Chinese word segmentation (Zhang and Clark, 2007; Jiang et al., 2008), and so on.

Similar to the situation in other sequence labeling problems, the training procedure is to learn a discriminative model mapping from inputs $x \in X$ to outputs $y \in Y$, where X is the set of sentences in the training corpus and Y is the set of corresponding labelled results. Following Collins, we use a function $\mathbf{GEN}(x)$ enumerating the candidate results of an input x , a representation $\mathbf{\Phi}$ mapping each training example $(x, y) \in X \times Y$ to a feature vector $\mathbf{\Phi}(x, y) \in R^d$, and a parameter vector $\vec{\alpha} \in R^d$ corresponding to the feature vector. For an input character sequence x, we aim to find an output $F(x)$ that satisfies:

$$F(x) = \underset{y \in \mathbf{GEN}(x)}{\text{argmax}} \; \mathbf{\Phi}(x, y) \cdot \vec{\alpha} \qquad (1)$$

where $\mathbf{\Phi}(x, y) \cdot \vec{\alpha}$ denotes the inner product of feature vector $\mathbf{\Phi}(x, y)$ and the parameter vector $\vec{\alpha}$.

Algorithm 1 depicts the pseudo code to tune the parameter vector $\vec{\alpha}$. In addition, the "averaged parameters" technology (Collins, 2002) is used to alleviate overfitting and achieve stable performance. Table 1 lists the feature template and corresponding instances. Following Ng and Low (2004), the current considering character is denoted as C_0, while the ith character to the left of C_0 as C_{-i}, and to the right as C_i. There are additional two functions of which each returns some property of a character. $Pu(\cdot)$ is a boolean function that checks whether a character is a punctuation symbol (returns 1 for a punctuation, 0 for not). $T(\cdot)$ is a multi-valued function, it classifies a character into four classifications: *number*, *date*, *English letter* and *others* (returns 1, 2, 3 and 4, respectively).

3 Automatic Annotation Adaptation

From this section, several shortened forms are adopted for representation inconvenience. We use *source corpus* to denote the corpus with the annotation standard that we don't require, which is of

Feature Template	Instances
C_i $(i = -2..2)$	C_{-2} = 九, C_{-1} = ○, C_0 = 年, C_1 = 代, C_2 = R
C_iC_{i+1} $(i = -2..1)$	$C_{-2}C_{-1}$ = 九○, $C_{-1}C_0$ = ○年, C_0C_1 = 年代, C_1C_2 = 代R
$C_{-1}C_1$	$C_{-1}C_1$ = ○代
$Pu(C_0)$	$Pu(C_0)$ = 0
$T(C_{-2})T(C_{-1})T(C_0)T(C_1)T(C_2)$	$T(C_{-2})T(C_{-1})T(C_0)T(C_1)T(C_2)$ = 11243

Table 1: Feature templates and instances from Ng and Low (Ng and Low, 2004). Suppose we are considering the third character "年" in "九○ 年 代R".

course the source of the adaptation, while *target corpus* denoting the corpus with the desired standard. And correspondingly, the two annotation standards are naturally denoted as *source standard* and *target standard*, while the classifiers following the two annotation standards are respectively named as *source classifier* and *target classifier*, if needed.

Considering that word segmentation and Joint S&T can be conducted in the same character classification manner, we can design an unified standard adaptation framework for the two tasks, by taking the source classifier's classification result as the guide information for the target classifier's classification decision. The following section depicts this adaptation strategy in detail.

3.1 General Adaptation Strategy

In detail, in order to adapt knowledge from the source corpus, first, a source classifier is trained on it and therefore captures the knowledge it contains; then, the source classifier is used to classify the characters in the target corpus, although the classification result follows a standard that we don't desire; finally, a target classifier is trained on the target corpus, with the source classifier's classification result as additional guide information. The training procedure of the target classifier automatically learns the regularity to transfer the source classifier's predication result from source standard to target standard. This regularity is incorporated together with the knowledge learnt from the target corpus itself, so as to obtain enhanced predication accuracy. For a given un-classified character sequence, the decoding is analogous to the training. First, the character sequence is input into the source classifier to obtain an source standard annotated classification result, then it is input into the target classifier with this classification result as additional information to get the final result. This coincides with the stacking method for combining dependency parsers (Martins et al., 2008; Nivre and McDonald, 2008), and is also similar to the Pred baseline for domain adaptation in (Daumé III and Marcu, 2006; Daumé III, 2007). Figures 2 and 3 show the flow charts for training and decoding.

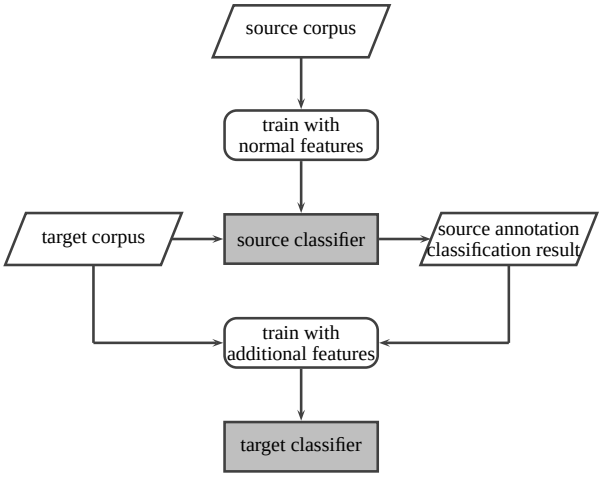

Figure 2: The pipeline for training.

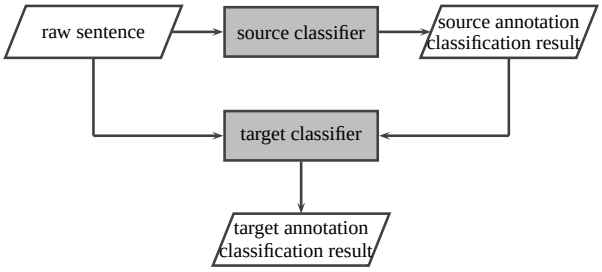

Figure 3: The pipeline for decoding.

The utilization of the source classifier's classification result as additional guide information resorts to the introduction of new features. For the current considering character waiting for classification, the most intuitive guide features is the source classifier's classification result itself. However, our effort isn't limited to this, and more special features are introduced: the source classifier's classification result is attached to every feature listed in Table 1 to get combined guide features. This is similar to feature design in discriminative dependency parsing (McDonald et al., 2005; Mc-

Donald and Pereira, 2006), where the basic features, composed of words and POSs in the context, are also conjoined with link direction and distance in order to obtain more special features. Table 2 shows an example of guide features and basic features, where "α = b " represents that the source classifier classifies the current character as *b*, the beginning of a word.

Such combination method derives a series of specific features, which helps the target classifier to make more precise classifications. The parameter tuning procedure of the target classifier will automatically learn the regularity of using the source classifier's classification result to guide its decision making. For example, if a current considering character shares some basic features in Table 2 and it is classified as *b*, then the target classifier will probably classify it as *m*. In addition, the training procedure of the target classifier also learns the relative weights between the guide features and the basic features, so that the knowledge from both the source corpus and the target corpus are automatically integrated together.

In fact, more complicated features can be adopted as guide information. For error tolerance, guide features can be extracted from n-best results or compacted lattices of the source classifier; while for the best use of the source classifier's output, guide features can also be the classification results of several successive characters. We leave them as future research.

Baseline Features
C_{-2} = 美
C_{-1} = 副
C_0 = 总
C_1 = 统
C_2 = 访
$C_{-2}C_{-1}$ = 美副
$C_{-1}C_0$ = 副总
C_0C_1 = 总统
C_1C_2 = 统访
$C_{-1}C_1$ = 副统
$Pu(C_0)$ = 0
$T(C_{-2})T(C_{-1})T(C_0)T(C_1)T(C_2)$ = 44444
Guide Features
α = b
C_{-2} = 美 $\circ$ α = b
C_{-1} = 副 $\circ$ α = b
C_0 = 总 $\circ$ α = b
C_1 = 统 $\circ$ α = b
C_2 = 访 $\circ$ α = b
$C_{-2}C_{-1}$ = 美副 $\circ$ α = b
$C_{-1}C_0$ = 副总 $\circ$ α = b
C_0C_1 = 总统 $\circ$ α = b
C_1C_2 = 统访 $\circ$ α = b
$C_{-1}C_1$ = 副统 $\circ$ α = b
$Pu(C_0)$ = 0 $\circ$ α = b
$T(C_{-2})T(C_{-1})T(C_0)T(C_1)T(C_2)$ = 44444 $\circ$ α = b

Table 2: An example of basic features and guide features of standard-adaptation for word segmentation. Suppose we are considering the third character "总" in "美副 总 统访华".

4 Related Works

Co-training (Sarkar, 2001) and classifier combination (Nivre and McDonald, 2008) are two technologies for training improved dependency parsers. The co-training technology lets two different parsing models learn from each other during parsing an unlabelled corpus: one model selects some unlabelled sentences it can confidently parse, and provide them to the other model as additional training corpus in order to train more powerful parsers. The classifier combination lets graph-based and transition-based dependency parsers to utilize the features extracted from each other's parsing results, to obtain combined, enhanced parsers. The two technologies aim to let two models learn from each other on the same corpora with the same distribution and annotation standard, while our strategy aims to integrate the knowledge in multiple corpora with different annotation-styles.

Gao et al. (2004) described a transformation-based converter to transfer a certain annotation-style word segmentation result to another style. They design some class-type transformation templates and use the transformation-based error-driven learning method of Brill (1995) to learn what word delimiters should be modified. However, this converter need human designed transformation templates, and is hard to be generalized to POS tagging, not to mention other structure labeling tasks. Moreover, the processing procedure is divided into two isolated steps, conversion after segmentation, which suffers from error propagation and wastes the knowledge in the corpora. On the contrary, our strategy is automatic, generalizable and effective.

In addition, many efforts have been devoted to manual treebank adaptation, where they adapt PTB to other grammar formalisms, such as such as CCG and LFG (Hockenmaier and Steedman, 2008; Cahill and Mccarthy, 2007). However, they are heuristics-based and involve heavy human engineering.

5 Experiments

Our adaptation experiments are conducted from People's Daily (PD) to Penn Chinese Treebank 5.0 (CTB). These two corpora are segmented following different segmentation standards and labeled with different POS sets (see for example Figure 1). PD is much bigger in size, with about 100K sentences, while CTB is much smaller, with only about 18K sentences. Thus a classifier trained on CTB usually falls behind that trained on PD, but CTB is preferable because it also annotates tree structures, which is very useful for downstream applications like parsing and translation. For example, currently, most Chinese constituency and dependency parsers are trained on some version of CTB, using its segmentation and POS tagging as the *de facto* standards. Therefore, we expect the knowledge adapted from PD will lead to more precise CTB-style segmenter and POS tagger, which would in turn reduce the error propagation to parsing (and translation).

Experiments adapting from PD to CTB are conducted for two tasks: word segmentation alone, and joint segmentation and POS tagging (Joint S&T). The performance measurement indicators for word segmentation and Joint S&T are *balanced F-measure*, $F = 2PR/(P+R)$, a function of *Precision* P and *Recall* R. For word segmentation, P indicates the percentage of words in segmentation result that are segmented correctly, and R indicates the percentage of correctly segmented words in gold standard words. For Joint S&T, P and R mean nearly the same except that a word is correctly segmented only if its POS is also correctly labelled.

5.1 Baseline Perceptron Classifier

We first report experimental results of the single perceptron classifier on CTB 5.0. The original corpus is split according to former works: chapters $271-300$ for testing, chapters $301-325$ for development, and others for training. Figure 4 shows the learning curves for segmentation only and Joint S&T, we find all curves tend to moderate after 7 iterations. The data splitting convention of other two corpora, People's Daily doesn't reserve the development sets, so in the following experiments, we simply choose the model after 7 iterations when training on this corpus.

The first 3 rows in each sub-table of Table 3 show the performance of the single perceptron models. Comparing row 1 and 3 in the sub-table below with the corresponding rows in the upper sub-table, we validate that when word segmentation and POS tagging are conducted jointly, the performance for segmentation improves since the POS tags provide additional information to word segmentation (Ng and Low, 2004). We also see that for both segmentation and Joint S&T, the performance sharply declines when a model trained on PD is tested on CTB (row 2 in each sub-table). In each task, only about 92% F_1 is achieved. This obviously fall behind those of the models trained on CTB itself (row 3 in each sub-table), about 97% F_1, which are used as the baselines of the following annotation adaptation experiments.

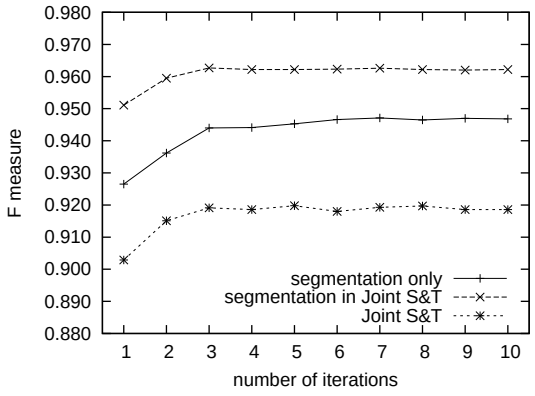

Figure 4: Averaged perceptron learning curves for segmentation and Joint S&T.

Train on	Test on	Seg F_1%	JST F_1%
Word Segmentation			
PD	PD	97.45	—
PD	CTB	91.71	—
CTB	CTB	97.35	—
PD → CTB	CTB	**98.15**	—
Joint S&T			
PD	PD	97.57	94.54
PD	CTB	91.68	—
CTB	CTB	97.58	93.06
PD → CTB	CTB	**98.23**	**94.03**

Table 3: Experimental results for both baseline models and final systems with annotation adaptation. PD → CTB means annotation adaptation from PD to CTB. For the upper sub-table, items of **JST** F_1 are undefined since only segmentation is performs. While in the sub-table below, **JST** F_1 is also undefined since the model trained on PD gives a POS set different from that of CTB.

POS	#Word	#BaseErr	#AdaErr	ErrDec%
AD	305	30	19	36.67 ↓
AS	76	0	0	
BA	4	1	1	
CC	135	8	8	
CD	356	21	14	33.33 ↓
CS	6	0	0	
DEC	137	31	23	25.81 ↓
DEG	197	32	37	↑
DEV	10	0	0	
DT	94	3	1	66.67 ↓
ETC	12	0	0	
FW	1	1	1	
JJ	127	41	44	↑
LB	2	1	1	
LC	106	3	2	33.33 ↓
M	349	18	4	77.78 ↓
MSP	8	2	1	50.00 ↓
NN	1715	151	126	16.56 ↓
NR	713	59	50	15.25 ↓
NT	178	1	2	↑
OD	84	0	0	
P	251	10	6	40.00 ↓
PN	81	1	1	
PU	997	0	1	↑
SB	2	0	0	
SP	2	2	2	
VA	98	23	21	08.70 ↓
VC	61	0	0	
VE	25	1	0	100.00 ↓
VV	689	64	40	37.50 ↓
SUM	6821	213	169	20.66 ↓

Table 4: Error analysis for Joint S&T on the developing set of CTB. **#BaseErr** and **#AdaErr** denote the count of words that can't be recalled by the baseline model and adapted model, respectively. **ErrDec** denotes the error reduction of *Recall*.

5.2 Adaptation for Segmentation and Tagging

Table 3 also lists the results of annotation adaptation experiments. For word segmentation, the model after annotation adaptation (row 4 in upper sub-table) achieves an F-measure increment of 0.8 points over the baseline model, corresponding to an error reduction of 30.2%; while for Joint S&T, the F-measure increment of the adapted model (row 4 in sub-table below) is 1 point, which corresponds to an error reduction of 14%. In addition, the performance of the adapted model for Joint S&T obviously surpass that of (Jiang et al., 2008), which achieves an F_1 of 93.41% for Joint S&T, although with more complicated models and features.

Due to the obvious improvement brought by annotation adaptation to both word segmentation and Joint S&T, we can safely conclude that the knowledge can be effectively transferred from on annotation standard to another, although using such a simple strategy. To obtain further information about what kind of errors be alleviated by annotation adaptation, we conduct an initial error analysis for Joint S&T on the developing set of CTB. It is reasonable to investigate the error reduction of *Recall* for each word cluster grouped together according to their POS tags. From Table 4 we find that out of 30 word clusters appeared in the developing set of CTB, 13 clusters benefit from the annotation adaptation strategy, while 4 clusters suffer from it. However, the compositive error rate of *Recall* for all word clusters is reduced by 20.66%, such a fact invalidates the effectivity of annotation adaptation.

Input Type	Parsing F_1%
gold-standard segmentation	82.35
baseline segmentation	80.28
adapted segmentation	**81.07**

Table 5: Chinese parsing results with different word segmentation results as input.

5.3 Contribution to Chinese Parsing

We adopt the Chinese parser of Xiong et al. (2005), and train it on the training set of CTB 5.0 as described before. To sketch the error propagation to parsing from word segmentation, we redefine the *constituent span* as a constituent subtree from a start character to a end character, rather than from a start word to a end word. Note that if we input the gold-standard segmented test set into the parser, the F-measure under the two definitions are the same.

Table 5 shows the parsing accuracies with different word segmentation results as the parser's input. The parsing F-measure corresponding to the gold-standard segmentation, 82.35, represents the "oracle" accuracy (i.e., upperbound) of parsing on top of automatic word segmention. After integrating the knowledge from PD, the enhanced word segmenter gains an F-measure increment of 0.8 points, which indicates that 38% of the error propagation from word segmentation to parsing is reduced by our annotation adaptation strategy.

6 Conclusion and Future Works

This paper presents an automatic annotation adaptation strategy, and conducts experiments on a classic problem: word segmentation and Joint

S&T. To adapt knowledge from a corpus with an annotation standard that we don't require, a classifier trained on this corpus is used to pre-process the corpus with the desired annotated standard, on which a second classifier is trained with the first classifier's predication results as additional guide information. Experiments of annotation adaptation from PD to CTB 5.0 for word segmentation and POS tagging show that, this strategy can make effective use of the knowledge from the corpus with different annotations. It obtains considerable F-measure increment, about 0.8 point for word segmentation and 1 point for Joint S&T, with corresponding error reductions of 30.2% and 14%. The final result outperforms the latest work on the same corpus which uses more complicated technologies, and achieves the state-of-the-art. Moreover, such improvement further brings striking F-measure increment for Chinese parsing, about 0.8 points, corresponding to an error propagation reduction of 38%.

In the future, we will continue to research on annotation adaptation for other NLP tasks which have different annotation-style corpora. Especially, we will pay efforts to the annotation standard adaptation between different treebanks, for example, from HPSG LinGo Redwoods Treebank to PTB, or even from a dependency treebank to PTB, in order to obtain more powerful PTB annotation-style parsers.

Acknowledgement

This project was supported by National Natural Science Foundation of China, Contracts 60603095 and 60736014, and 863 State Key Project No. 2006AA010108. We are especially grateful to Fernando Pereira and the anonymous reviewers for pointing us to relevant domain adaption references. We also thank Yang Liu and Haitao Mi for helpful discussions.

References

Daniel M. Bikel and David Chiang. 2000. Two statistical parsing models applied to the chinese treebank. In *Proceedings of the second workshop on Chinese language processing*.

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proceedings of EMNLP*.

Eric Brill. 1995. Transformation-based error-driven learning and natural language processing: a case study in part-of-speech tagging. In *Computational Linguistics*.

Sabine Buchholz and Erwin Marsi. 2006. Conll-x shared task on multilingual dependency parsing. In *Proceedings of CoNLL*.

Aoife Cahill and Mairead Mccarthy. 2007. Automatic annotation of the penn treebank with lfg f-structure information. In *in Proceedings of the LREC Workshop on Linguistic Knowledge Acquisition and Representation: Bootstrapping Annotated Language Data*.

David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, pages 201–228.

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

Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the Empirical Methods in Natural Language Processing Conference*, pages 1–8, Philadelphia, USA.

Hal Daumé III and Daniel Marcu. 2006. Domain adaptation for statistical classifiers. In *Journal of Artificial Intelligence Research*.

Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *Proceedings of ACL*.

Jianfeng Gao, Andi Wu, Mu Li, Chang-Ning Huang, Hongqiao Li, Xinsong Xia, and Haowei Qin. 2004. Adaptive chinese word segmentation. In *Proceedings of ACL*.

Julia Hockenmaier and Mark Steedman. 2008. Ccgbank: a corpus of ccg derivations and dependency structures extracted from the penn treebank. In *Computational Linguistics*, volume 33(3), pages 355–396.

Wenbin Jiang, Liang Huang, Yajuan Lü, and Qun Liu. 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*.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank. In *Computational Linguistics*.

André F. T. Martins, Dipanjan Das, Noah A. Smith, and Eric P. Xing. 2008. Stacking dependency parsers. In *Proceedings of EMNLP*.

Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proceedings of EACL*, pages 81–88.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of ACL*, pages 91–98.

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 Empirical Methods in Natural Language Processing Conference*.

Joakim Nivre and Ryan McDonald. 2008. Integrating graph-based and transition-based dependency parsers. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*.

Stephan Oepen, Kristina Toutanova, Stuart Shieber, Christopher Manning Dan Flickinger, and Thorsten Brants. 2002. The lingo redwoods treebank: Motivation and preliminary applications. In *In Proceedings of the 19th International Conference on Computational Linguistics (COLING 2002)*.

Anoop Sarkar. 2001. Applying co-training methods to statistical parsing. In *Proceedings of NAACL*.

Fei Xia. 2000. The part-of-speech tagging guidelines for the penn chinese treebank (3.0). In *Technical Reports*.

Deyi Xiong, Shuanglong Li, Qun Liu, and Shouxun Lin. 2005. Parsing the penn chinese treebank with semantic knowledge. In *Proceedings of IJCNLP 2005*, pages 70–81.

Nianwen Xue and Libin Shen. 2003. Chinese word segmentation as lmr tagging. In *Proceedings of SIGHAN Workshop*.

Nianwen Xue, Fei Xia, Fu-Dong Chiou, and Martha Palmer. 2005. The penn chinese treebank: Phrase structure annotation of a large corpus. In *Natural Language Engineering*.

Shiwen Yu, Jianming Lu, Xuefeng Zhu, Huiming Duan, Shiyong Kang, Honglin Sun, Hui Wang, Qiang Zhao, and Weidong Zhan. 2001. Processing norms of modern chinese corpus. Technical report.

Yue Zhang and Stephen Clark. 2007. Chinese segmentation with a word-based perceptron algorithm. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*.