Learning Reliable Information for Dependency Parsing Adaptation

Wenliang Chen[†], **Youzheng Wu**[‡], **Hitoshi Isahara**[*]
[†]Language Infrastructure Group
[‡]Spoken Language Communication Group, ATR
[*]Machine Translation Group
National Institute of Information and Communications Technology
3-5 Hikari-dai, Seika-cho, Soraku-gun, Kyoto, Japan, 619-0289
{chenwl, youzheng.wu, isahara}@nict.go.jp

Abstract

In this paper, we focus on the adaptation problem that has a large labeled data in the source domain and a large but unlabeled data in the target domain. Our aim is to learn reliable information from unlabeled target domain data for dependency parsing adaptation. Current state-of-the-art statistical parsers perform much better for shorter dependencies than for longer ones. Thus we propose an adaptation approach by learning reliable information on shorter dependencies in an unlabeled target data to help parse longer distance words. The unlabeled data is parsed by a dependency parser trained on labeled source domain data. The experimental results indicate that our proposed approach outperforms the baseline system, and is better than current state-of-the-art adaptation techniques.

1 Introduction

Dependency parsing aims to build the dependency relations between words in a sentence. There are many supervised learning methods for training high-performance dependency parsers(Nivre et al., 2007), if given sufficient labeled data. However, the performance of parsers declines when we are in the situation that a parser is trained in one "source" domain but is to parse the sentences in a second "target" domain. There are two tasks(Daumé III, 2007) for the domain adaptation problem. The first one is that we have a large labeled data in the source domain and a small labeled data in target domain. The second is similar, but instead of having a small labeled target data, we have a large but unlabeled target data. In this paper, we focus on the latter one.

Current statistical dependency parsers perform worse while the distance of two words is becoming longer for domain adaptation. An important characteristic of parsing adaptation is that the parsers perform much better for shorter dependencies than for longer ones (the score at length l is much higher than the scores at length$> l$).

In this paper, we propose an approach by using the information on shorter dependencies in auto-parsed target data to help parse longer distance words for adapting a parser. Compared with the adaptation methods of Sagae and Tsujii (2007) and Reichart and Rappoport (2007), our approach uses the information on word pairs in auto-parsed data instead of using the whole sentences as newly labeled data for training new parsers. It is difficult to detect reliable parsed sentences, but we can find relative reliable parsed word pairs according to dependency length. The experimental results show that our approach significantly outperforms baseline system and current state of the art techniques.

2 Motivation and prior work

In dependency parsing, we assign head-dependent relations between words in a sentence. A simple example is shown in Figure 1, where the arc between *a* and *hat* indicates that *hat* is the head of *a*.

Current statistical dependency parsers perform better if the dependency lengthes are shorter (McDonald and Nivre, 2007). Here the length of the dependency from word w_i to word w_j is simply equal to $|i - j|$. Figure 2 shows the results (F_1

Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008), pages 113–120
Manchester, August 2008

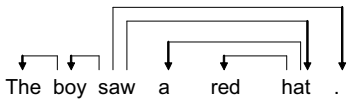

Figure 1: An example for dependency relations.

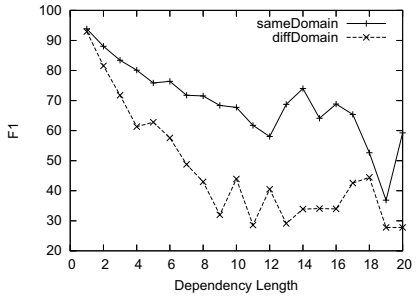

Figure 2: The scores relative to dependency length. "SameDomain" refers to training and testing in the same domain, and "diffDomain" refers to training and testing in two domains (domain adaptation).

score)[1] on our testing data, provided by a deterministic parser, which is trained on labeled source data. Comparing two curves at the figure, we find that the scores of diffDomain decreases much more sharply than the scores of sameDomain, when dependency length increases. The score decreases from about 92% at length 1 to 50% at 7. When lengthes are larger than 7, the scores are below 50%. We also find that the score at length l is much higher (around 10%) than the score at length $l + 1$ from length 1 to 7. There is only one exception that the score at length 4 is a little less than the score at length 5. But this does not change so much and the scores at length 4 and 5 are much higher than the one at length 6.

Two words (word w_i and word w_j) having a dependency relation in one sentence can be adjacent words (word distance = 1), neighboring words (word distance = 2), or the words with distance > 2 in other sentences. Here the distance of word pair (word w_i and word w_j) is equal to $|i - j|$. For example, "a" and "hat" has dependency relation in the sentence at Figure 1. They can also be adjacent words in the sentence "The boy saw a hat." and the words with distance = 3 in "I see a red beautiful hat.". This makes it possible for the word pairs with different distances to share the information.

According to the above observations, we present an idea that *the information on shorter dependencies in auto-parsed target data is reliable for parsing the words with longer distance* for domain adaptation. Here, "shorter" is not exactly short. That is to say, the information on dependency length l in auto-parsed data can be used to help parse the words whose distances are longer than l when testing, where l can be any number. We do not use the dependencies whose lengthes are too long because the accuracies of long dependencies are very low.

In the following content, we demonstrate our idea with an example. The example shows how to use the information on length 1 to help parse two words whose distance is longer than 1. Similarly, the information on length l can also be used to help parse the words whose distance is longer than l.

Figure 2 shows that the dependency parser performs best at tagging the relations between adjacent words. Thus, we expect that dependencies of adjacent words in auto-parsed target data can provide useful information for parsing words whose distances are longer than 1. We suppose that our task is Chinese dependency parsing adaptation.

Here, we have two words "大型JJ(large-scale)" and "展 览NN(exhibition)". Figure 3 shows the examples in which word distances of these two words are different. For the sentences in the bottom part, there is a ambiguity of "JJ + NN1 + NN2" at "大 型JJ(large-scale)/艺 术NN(art)/展 览NN(exhibition)", "大 型JJ(large-scale)/文 化NN(culture)/艺 术NN(art)/展 览NN(exhibition)" and "大 型JJ(large-scale)/中 国NR(China)/文 化NN(culture)/艺 术NN(art)/展 览NN(exhibition)". Both NN1 and NN2 could be the head of JJ. In the examples in the upper part, "大 型JJ(large-scale)" and "展 览NN(exhibition)" are adjacent words, for which current parsers can work well. We use a parser to parse the sentences in the upper part. "展览(exhibition)" is assigned as the head of "大型(large-scale)". Then we expect the information from the upper part can help parse the sentences in the bottom part. Now, we consider what a learning model could do to assign the appropriate relation between "大 型(large-scale)" and "展 览(exhibition)" in the bottom part. We provide additional information that "展览(exhibition)" is the possible head of "大 型(large-scale)" in the auto-parsed data (the upper part). In this way, the learning model may use this information to make correct decision.

[1] $F_1 = 2 \times precision \times recall/(precision + recall)$ where precision is the percentage of predicted arcs of length d that are correct and recall is the percentage of gold standard arcs of length d that are correctly predicted.

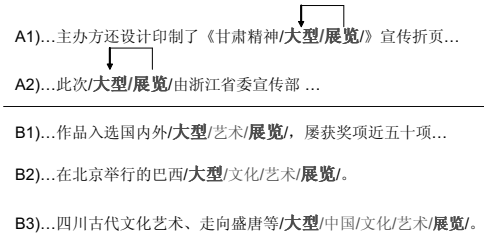

Figure 3: Examples for "大型(large-scale)" and "展览(exhibition)". The upper part (A) refers to the sentences from unlabeled data and the bottom part (B) refers to the sentences waiting for parsing.

Up to now, we demonstrate how to use the information on length 1. Similarly, we can use the information on length 2, 3, ... By this way, we propose an approach by exploiting the information from a large-scale unlabeled target data for dependency parsing adaptation.

In this paper, our approach is to use unlabeled data for parsing adaptation. There are several studies relevant to ours as described below.

CoNLL 2007(Nivre et al., 2007) organized a shared task for domain adaptation without annotated data in new domain. The labeled data was from the Wall Street Journal, the development data was from biomedical abstracts, and the testing data was from chemical abstracts and parent-child dialogues. Additionally, a large unlabeled corpus was provided. The systems by Sagae and Tsujii (2007), Attardi et al. (2007), and Dredze et al. (2007) performed top three in the shared task.

Sagae and Tsujii (2007) presented a procedure similar to a single iteration of co-training. Firstly, they trained two parsers on labeled source data. Then the two parsers were used to parse the sentences in unlabeled data. They selected only identical parsing results produced by the two parsers. Finally, they retrained a parser on newly parsed sentences and the original labeled data. They performed the highest scores for this track. Attardi et al. (2007) presented a procedure with correcting errors by a revision techniques. Dredze et al. (2007) submitted parsing results without adaptation. They declared that it was difficult to significantly improve performance on any test domain beyond that of a state-of-the-art parser. Their error analysis suggested that the primary cause of loss from adaptation is because of differences in the annotation guidelines. Without specific knowledge of the target domain's annotation standards, significant improvement can not be made.

Reichart and Rappoport (2007) studied self-training method for domain adaptation (The WSJ data and the Brown data) of phrase-based parsers. McClosky et al. (2006) presented a successful instance of parsing with self-training by using a reranker. Both of them used the whole sentences as newly labeled data for adapting the parsers, while our approach uses the information on word pairs.

Chen et al. (2008) presented an approach by using the information of adjacent words for in-domain parsing. As Figure 2 shows, the score curves of sameDomain (in-domain) parsing and diffDomain (out-domain) parsing are quite different. Our work focuses on parsing adaptation and is based on the fact that current parsers perform much better for shorter dependencies than for longer ones. This causes that our work differs in that we use the information on shorter dependencies in auto-parsed target data to help parse the words with longer distance for parsing adaptation. In this paper, "shorter" and "longer" are relative. Length l is relatively shorter than length $l + 1$, where l can be any number.

3 The parsing approach

In this paper, we choose the model described by Nivre (2003) as our parsing model. It is a deterministic parser and works quite well in the shared-task of CoNLL2006(Nivre et al., 2006).

3.1 The parsing model

The Nivre (2003) model is a shift-reduce type algorithm, which uses a stack to store processed tokens and a queue to store remaining input tokens. It can perform dependency parsing in O(n) time. The dependency parsing tree is built from atomic actions in a left-to-right pass over the input. The parsing actions are defined by four operations: Shift, Reduce, Left-Arc, and Right-Arc, for the stack and the queue. TOP is the token on top of the stack and NEXT is next token in the queue. The Left-Arc and Right-Arc operations mean that there is a dependency relation between TOP and NEXT.

The model uses a classifier to produce a sequence of actions for a sentence. In this paper, we use the SVM model. And LIBSVM(Chang and Lin, 2001) is used in our experiments.

Note that the approach (see section 4)we present in this paper can also be applied to

other parsers, such as the parser by Yamada and Matsumoto (2003), or the one by McDonald et al. (2006).

3.2 Parsing with basic features

The parser is a history-based parsing model, which relies on features of the parsed tokens to predict next parsing action. We represent basic features based on words and part-of-speech (POS) tags. The basic features are listed as follows:

- Lexical Features on TOP: the word of TOP, the word of the head of TOP, and the words of leftmost and rightmost dependent of TOP.
- Lexical Features on NEXT: the word of NEXT and the word of the token immediately after NEXT in the original input string.
- POS features on TOP: the POS of TOP, the POS of the token immediately below TOP, and the POS of leftmost and rightmost dependent of TOP.
- POS features on NEXT: the POS of NEXT, the POS of next three tokens after NEXT, and the POS of the token immediately before NEXT in original input string.

Based on the above parsing model and basic features, we train a basic parser on annotated source data. In the following content, we call this parser Basic Parser.

4 Domain adaptation with shorter dependency

This section presents our adaptation approach by using the information based on relative shorter dependencies in auto-parsed data to help parse the words whose distances are longer. Firstly, we use the Basic Parser to parse all the sentences in unlabeled target data. Then we explore reliable information based on dependency relations in auto-parsed data. Finally, we incorporate the features based on reliable information into the parser to improve performance.

4.1 Extracting word pairs from auto-parsed data

In this section, we collect word pairs from the auto-parsed data. At first, we collect the word pairs with length 1. In a parsed sentence, if two words have dependency relation and their word distance is 1, we will add this word pair into the list L_{dep} and count its frequency. We also consider the direction, LA for left arc and RA for right arc. For example, "大型(large-scale)" and "展览(exhibition)" are adjacent words in the sentence "我们(We)/举办(held)/大型(large-scale)/展览(exhibition)/。" and have a left dependency arc assigned by the Basic Parser. The word pair "大型(large-scale)-展览(exhibition)" with "LA" is added into L_{dep}.

Similarly, we collect the pairs whose word distances are longer than 1. In L_{dep}, with length l and direction dr(LA or RA), the pair p_u has $freq_l(p_u : dr)$. For example, $freq_2(p_u : LA) = 3$ refers to the word pair p_u with left arc(LA) occurs 3 times in the auto-parsed data when two words' distance is 2. Because figure 2 shows that the accuracies of long dependencies are low, we only collect the pairs whose distances are not larger than a predefined length l_{max}.

4.2 The adaptation approach

The word pair p_t is the pair $< w_i, w_j >$.

4.2.1 The information on shorter distances

If the distance of p_t is d, we will use the pairs whose lengthes are less than d. It results in the words with different distances using different set of word pairs in L_{dep}. For example, if d is 5, we can use the pairs with dependency lengthes from 1 to 4 in L_{dep}. The information is represented by the equation as follows:

$$I_d(p_t : dr) = \begin{cases} 0 & p_t \notin L_{dep} \\ freq_1(p_t : dr) & d = 1 \\ \sum_{l=1}^{d-1} freq_l(p_t : dr) & d > 1 \end{cases} \quad (1)$$

4.2.2 Classifying into buckets

According to $I_d(p_t : dr)$, word pairs are grouped into different buckets as follows:

$$Bucket_d(p_t : dr) = \begin{cases} B_0 & I_d(p_t : dr) = 0 \\ B_1 & 0 < I_d(p_t : dr) \le f_1 \\ \dots & \\ B_n & f_{n-1} < I_d(p_t : dr) \le f_n \\ B_a & f_n < I_d(p_t : dr) \end{cases} \quad (2)$$

where, $f_1, f_2, ..., f_n$ are the thresholds. For example, I_3(大型-展览:LA) is 20, $f_3 = 15$ and $f_4 = 25$. Then it is grouped into the bucket B_4. We set $f_1 = 2$, $f_2 = 8$, and $f_3 = 15$ in the experiments.

4.2.3 Parsing with the adapting Features

Based on the buckets of word pairs, we represent new features on labeled source data for the parser. We call these new features adapting features. According to different word distances between TOP and NEXT, the features are listed at Table 1. So we have 8 types of the adapting features, including 2 types for distance=1, 3 types for distance=2, and 3 types for distance≥3. Each feature is formatted as "DistanceType:FeatureType:Bucket", where DistanceType is D1, D2, or D3 corresponding to

three distances, FeatureType is FB0, FB1, or FB_1 corresponding to three positions. Here, if a word pair has two dependency directions in L_{dep}, we will choose the direction having higher frequency.

Then using the parsing model of Nivre (2003), we train a new parser based on the adapting features and basic features.

distance	FB_1	FB0	FB1
=1		+	+
=2	+	+	+
≥3	+	+	+

Table 1: Adapting features. FB0 refers to the bucket of the word pair of TOP and NEXT, FB1 refers to the bucket of the word pair of TOP and next token after NEXT, and FB_1 refers to the bucket of the word pair of TOP and the token immediately before NEXT. "+" refers to this item having this type of feature.

4.2.4 An example

We show an example for representing the adapting features. For example, we have the string "大 型JJ(large-scale)/文 化NN(culture)/艺术NN(art)/展 览NN(exhibition)/。". And "大型(large-scale)" is TOP and "展览(exhibition)" is NEXT. Because the distance of TOP and NEXT is 3, we have three features. We suppose that (FB0) the bucket of the word pair ("大型-展览") of TOP and NEXT is bucket B_4, (FB1) the bucket of the word pair ("大型-。") of TOP and next token after NEXT is bucket B_0, and (FB_1) the bucket of the word pair ("大型-艺术")of TOP and the token immediately before NEXT is bucket B_1. Then, we have the features: "D3:FB0:B_4", "D3:FB1:B_0", and "D3:FB_1:B_1".

4.3 Adaptation for unknown word[2]

The unknown word problem is an important issue for domain adaptation(Dredze et al., 2007). Our approach can work for improving performance of parsing unknown word pairs in which there is at least one unknown word. We collect word pairs including unknown word pairs at Section 4.1. Then unknown word pairs in testing data are also mapped into one of the buckets via Equation (2). So known word pairs can share the features with unknown word pairs.

[2]An unknown word is a word that is not included in training data.

5 Experimental setup

CoNLL 2007(Nivre et al., 2007) organized the domain adaptation task and provided a data set in English. However, the data set had differences between the annotation guidelines in source and target domains. Without specific knowledge of the target domain's annotation standards, significant improvement can not be made(Dredze et al., 2007). In this paper, we discussed the situation that the data of source and target domains were annotated under the same annotation guideline. So we used a data set converted from Penn Chinese Treebank (CTB)[3].

Labeled data: the CTB(V5.0) was used in our experiments. The data set was converted by the same rules for conversion as Chen et al. (2008) did. We used files 1-270, 400-554, and 600-931 as source domain training data (STrain), files 271-300 as source domain testing data (STest) and files 590-596 as target domain testing data (TTest). We used the gold standard segmentation and POS tags in the CTB. The target domain data was from Sinorama magazine, Taiwan and the source domain data was mainly from Xinhua newswire, mainland of China. The genres of these two parts were quite different. Table 2 shows the statistical information of the data sets. Given the words of the STrain data, TTest included 30.79% unknown words. We also checked the distribution of POS tags. The difference was large, too.

Unlabeled data: three data sets were used in our experiments, including the PFR data (5.44M words), the CKIP data (5.44M words), and the SINO data (25K words). The PFR corpus[4] included the documents from People's Daily at 1998 and we used about 1/3 of all sentences. The CKIP[5] corpus was used for SIGHAN word segmentation bakeoff 2005. To simplify, we used their segmentation. The SINO data was the files 1001-1151 of CTB, also from Sinorama magazine, the same as our testing target data. We removed the annotation tags from the SINO data. Among the three unlabeled data, the SINO data was closest to testing target data because they came from the same resource. Table 2 lists the information of data sets. From the table, we found that the PFR data was

[3]More detailed information can be found at http://www.cis.upenn.edu/~chinese/.

[4]More detailed information can be found at http://www.icl.pku.edu.

[5]More detailed information can be found at http://rocling.iis.sinica.edu.tw/CKIP/index.htm

	Num Of Words	Unknown Word Rate
STrain	17983	-
STest	1829	9.73
TTest	1783	30.79
CKIP	140k	-
STest	1829	11.42
TTest	1783	8.63
PFR	123k	-
STest	1829	8.58
TTest	1783	15.64

Table 2: The information of the data sets

closer to source domain and the CKIP data was closer to target domain.

To assign POS tags for the unlabeled data, we used the package TNT (Brants, 2000) to train a POS tagger on training data. Because the PFR data and the CTB used different POS standards, we did not use the POS tags in the PFR data.

We measured the quality of the parser by the unlabeled attachment score (UAS), i.e., the percentage of tokens with correct head. We also reported the accuracy of ROOT.

6 Results

In the following content, OURS refers to our proposed approach. The baseline system refers to the Basic Parser.

6.1 Basic experiments

In this section, we examined the performance of baseline systems and our proposed approach with different unlabeled data sets.

Table 3 shows the experimental results, where "OURS with SINO(GOLD)" refers to the parser using gold standard POS tags, and "OURS with SINO(AUTO)" refers to the parser using auto-assigned POS tags. From the two results of baseline, we found that the parser performed very differently in two domains by 8.24%.

With the help of SINO(AUTO), OURS provided 1.11% improvement for UAS and 6.16% for ROOT. If we used gold standard POS tags, the score was 78.40% for UAS (1.34% improvement), and 65.40% for ROOT (6.64% improvement). By using the SINO data, our approach achieved significant improvements over baseline system. It was surprised that OURS with CKIP achieved 78.30% score, just a little lower than the one with SINO(GOLD). The reason may be that the size of the CKIP data was much bigger than the SINO data. So we can obtain more word pairs from the CKIP data. The parser achieved 0.30% improvement with PFR. Even though the size of the SINO data was smaller, the parser performed well with its help.

Data	UAS	ROOT
baseline(STest)	85.30	88.21
baseline(TTest)	77.06	58.76
OURS with SINO(GOLD)	78.40(+1.34)	65.40
OURS with SINO(AUTO)	78.17(+1.11)	64.92
OURS with CKIP	78.30(+1.24)	65.87
OURS with PFR	77.36(+0.30)	63.03

Table 3: Basic results

l_{max}	SINO(GOLD)	SINO(AUTO)
1	77.84	77.80
3	78.03	77.95
5	78.22	**78.17**
7	**78.40**	78.11
9	78.38	78.13
∞	78.35	78.09

Table 4: The effect of different l_{max}

These results indicated that we should collect the unlabeled data that is closer to target domain or larger. The improvements of OURS with CKIP and OURS with SINO were significant in one-tail paired t-test ($p < 10^{-5}$).

6.2 The effect of different l_{max}

Table 4 shows the experimental results, where l_{max} is described at Section 4.1. With SINO(GOLD), our parser performed best at $l_{max} = 7$. And with SINO(AUTO), it performed best at $l_{max} = 5$. These indicated that our approach can incorporate pairs with different lengthes to improve performance. We also found that the long dependencies were not reliable, as the curve (diffDomain) of Figure 2 showed that the scores were less than 50% when lengthes were larger than 8.

6.3 Comparison of other systems

In this section, we turned to compare our approach with other methods. We implemented two systems: SelfTrain and CoTrain. The SelfTrain system was following to the method described by Reichart and Rappoport (2007) and randomly selected new auto-parsed sentences. The CoTrain system was similar to the learning scheme described by Sagae and Tsujii (2007). However, we did not use the same parsing algorithms as the ones used by Sagae and Tsujii (2007). Firstly, we

Method	UAS	ROOT
baseline	77.06	58.76
SelfTrain	77.44	60.18
CoTrain	77.57	60.81
OURS	78.30	65.87

Table 5: The results of several adaptation methods with CKIP

trained a forward parser (same as our baseline system) and a backward parser. Then the identical parsed sentences by the two parsers were selected as newly labeled data. Finally, we retrained a forward parser with new training data. We selected the sentences having about 200k words from the CKIP data as newly labeled data for the SelfTrain and CoTrain systems.

Table 5 shows the experimental results. Both systems provided about 0.4%-0.5% improvement over baseline system. Our approach performed best among all systems. Another problem was that the time for training the SelfTrain and CoTrain systems became much longer because they almost used double size of training data.

7 Analysis

In this section, we try to understand the benefit in our proposed adaptation methods. Here, we compare OURS's results with baseline's.

7.1 Improvement relative to dependency length

We presented an idea that using the information on shorter dependencies in auto-parsed target data to help parse the words with longer distance for domain adaptation. In this section, we investigated how our approach performed for parsing longer distance words. Figure 4 shows the improvement relative to dependency length. From the figure, we found that our approach always performed better than baseline when dependency lengthes were 1-7. Especially, our approach achieved improvements by 2.58% at length 3, 5.38% at 6, and 3.67% at 7. For longer ones, the improvement was not stable. One reason may be that the numbers of longer ones were small. Another reason was that parsing long distance words was very difficult. However, we still found that our approach did improve the performance for longer ones, by performing better at 8 points and worse at 5 points when length was not less than 8.

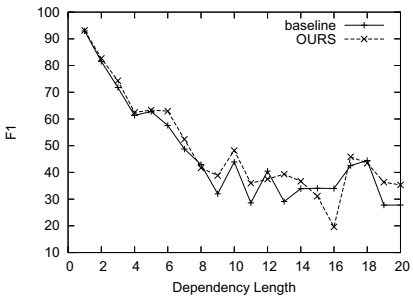

Figure 4: Performance as a function of dependency length

7.2 Improvement relative to unknown words

The unknown word problem is an important issue for adaptation. Our approach can partially release the unknown word problem. We listed the data of the numbers of unknown words from 0 to 8 because the number of sentences was very small for others. We grouped each sentence into one of three classes: (Better) those where our approach's score increased relative to the baseline's score, (NoChange) those where the score remained the same, and (Worse) those where the score had a relative decrease. We added another class (NoWorse) by merging Better and NoChange.

Figure 5 shows the experimental results, where x axis refers to the number of unknown words in one sentence and y axis refers to how many percent the class has. For example, for the sentences having 5 unknown words, about 45.45% improved, 22.73% became worse, 31.82% kept unchanged, and 77.27% did not become worse. The NoWorse curve showed that regardless of the number of unknown words in a sentence, there was more than 60% chance that our approach did not harm the result. The Better curve and Worse curve showed that our approach always provided better results. Our approach achieved most improvement for the middle ones. The reason was that parsing the sentence having too many unknown words was very difficult.

7.3 Improvement relative to POS pairs

In this section, we listed the improvements relative to POS tags of paired words having a dependency relation. Table 6 shows the accuracies of baseline and OURS on TOP 20 POS pairs (ordered by the frequencies of their occurrences in testing data), where "A1" refers to the accuracy of baseline, "A2" refers to the accuracy of OURS, and "Pairs" is the POS pairs of dependent-head.

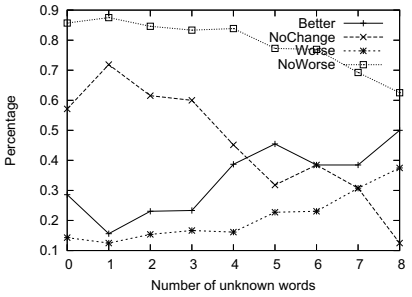

Figure 5: Performance as a function of number of unknown words

Pairs	A1	A2(A2-A1)	Pairs	A1	A2(A2-A1)
NN-VV	79.61	81.90(+2.29)	DEG-NN	94.74	94.74(=)
VV-VV	50.00	50.40(+0.40)	CD-M	96.77	97.85(+1.08)
NN-NN	86.08	87.26(+1.18)	NN-P	76.92	76.92(=)
AD-VV	91.01	91.01(=)	JJ-NN	92.11	94.74(+2.63)
P-VV	68.60	70.25(+1.65)	AD-VA	98.55	98.55(=)
DEC-NN	97.48	98.32(+0.84)	NN-VA	78.95	84.21(+5.26)
NR-VV	81.98	81.98(=)	NN-DEG	96.43	94.64(-1.79)
VV-DEC	74.07	73.15(-0.92)	VV-VC	40.82	46.94(+6.12)
NR-NN	87.85	87.85(=)	AD-VC	95.92	93.88(-2.04)
NN-VC	90.91	91.92(+1.01)	VA-VV	60.87	67.39(+6.52)

Table 6: Improvement relative to POS pairs

For example, "NN-VV" means that "NN" is the POS of the dependent and "VV" is the POS of the head. And baseline yielded 79.61% accuracy and OURS yielded 81.90% (2.29% higher) on "NN-VV". From the table, we found that our approach worked well for most POS pairs (better for eleven pairs, no change for six, and worse for three).

8 Conclusion

This paper presents a simple but effective approach to adapt dependency parser by using unlabeled target data. We extract the information on shorter dependencies in an unlabeled data parsed by a basic parser to help parse longer distance words. The experimental results show that our approach significantly outperforms baseline system and current state of the art adaptation techniques.

There are a lot of ways in which this research could be continued. First, we can apply our approach to other languages because our approach is independent on language. Second, we can enlarge the unlabeled data set to obtain more word pairs to provide more information for the parsers.

References

Attardi, Giuseppe, Felice Dell'Orletta, Maria Simi, Atanas Chanev, and Massimiliano Ciaramita. 2007. Multilingual dependency parsing and domain adaptation using DeSR. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 1112–1118.

Brants, T. 2000. TnT–a statistical part-of-speech tagger. *Proceedings of the 6th Conference on Applied Natural Language Processing*, pages 224–231.

Chang, C.C. and C.J. Lin. 2001. LIBSVM: a library for support vector machines. *Software available at http://www. csie. ntu. edu. tw/cjlin/libsvm*.

Chen, W., D. Kawahara, K. Uchimoto, Y. Zhang, and H. Isahara. 2008. Dependency parsing with short dependency relations in unlabeled data. In *Proceedings of IJCNLP 2008*.

Daumé III, Hal. 2007. Frustratingly easy domain adaptation. In *Proceedings of ACL 2007*, Prague, Czech Republic.

Dredze, Mark, John Blitzer, Partha Pratim Talukdar, Kuzman Ganchev, João Graca, and Fernando Pereira. 2007. Frustratingly hard domain adaptation for dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 1051–1055.

McClosky, D., E. Charniak, and M. Johnson. 2006. Reranking and self-training for parser adaptation. In *Proceedings of Coling-ACL*, pages 337–344.

McDonald, Ryan and Joakim Nivre. 2007. Characterizing the errors of data-driven dependency parsing models. In *Proceedings of EMNLP-CoNLL*, pages 122–131.

McDonald, Ryan, Kevin Lerman, and Fernando Pereira. 2006. Multilingual dependency analysis with a two-stage discriminative parser. In *Proceedings of CoNLL-X*, New York City, June.

Nivre, J., J. Hall, J. Nilsson, G. Eryigit, and S Marinov. 2006. Labeled pseudo-projective dependency parsing with support vector machines. In *CoNLL-X*.

Nivre, Joakim, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932.

Nivre, J. 2003. An efficient algorithm for projective dependency parsing. In *Proceedings of IWPT2003*, pages 149–160.

Reichart, Roi and Ari Rappoport. 2007. Self-training for enhancement and domain adaptation of statistical parsers trained on small datasets. In *Proceedings of ACL*, Prague, Czech Republic, June.

Sagae, Kenji and Jun'ichi Tsujii. 2007. Dependency parsing and domain adaptation with LR models and parser ensembles. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 1044–1050.

Yamada, H. and Y. Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of IWPT2003*, pages 195–206.