Language Resource Addition Strategies for Raw Text Parsing

Atsushi Ushiku[1], **Tetsuro Sasada**[2], **Shinsuke Mori**[3]
[1]Graduate School of Informatics, Kyoto University
[2,3]Academic Center for Computing and Media Studies, Kyoto University
[1,2,3]Yoshidahonmachi, Sakyo-ku, Kyoto, Japan

Abstract

We focus on the improvement of accuracy of raw text parsing, from the viewpoint of language resource addition. In Japanese, the raw text parsing is divided into three steps: word segmentation, part-of-speech tagging, and dependency parsing. We investigate the contribution of language resource addition in each of three steps to the improvement in accuracy for two domain corpora. The experimental results show that this improvement depends on the target domain. For example, when we handle well-written texts of limited vocabulary, white paper, an effective language resource is a word-POS pair sequence corpus for the parsing accuracy. So we conclude that it is important to check out the characteristics of the target domain and to choose a suitable language resource addition strategy for the parsing accuracy improvement.

Keywords: language resource addition, part-of-speech tagging, dependency parsing

1. Introduction

For languages without clear word boundary natural language processing (NLP) , such as word segmentation (WS), part-of-speech (POS) tagging (PT), and dependency parsing (DP) is essential technology for applications such as machine translation or information extraction (Matsumoto et al., 2000; Kudo et al., 2004; Kurohashi et al., 1994; Neubig et al., 2011; Kurohashi and Nagao, 1994; Mori et al., 2000; Kudo and Matsumoto, 2002; Flannery et al., 2012). WS and PT are now sufficiently accurate and widely used in various applications. DP is, however, not so widely used as WS and PT because of its insufficient performance. Recently there are many attempts at using sentence structure such as tree-based machine translation, language generation, etc. Thus the accuracy of a dependency parser is getting more and more important.

In this paper, we discuss DP from raw sentences in Japanese. Although Japanese does not have clear word boundary, we take words as the DP unit similar to NLP in other languages such as English (Buchholz and Marsi, 2006; McDonald and Pereira, 2006; McDonald et al., 2005; Nivre and Nilsson, 2005; Koo and Collins, 2010; Oflazer, 1999; McDonald et al., 2006; Sagae and Lavie, 2006; Eisner, 1996; Nivre and Scholz, 2004), and we adopt a three-step pipeline method consisting of a word segmenter, a POS tagger, and a dependency parser (Figure 1). Although some researchers have proposed joint approaches (Hatori et al., 2012), the pipeline method has advantages that it can utilize various types of language resources easily and it is easy to analyze the effects of each step. So we can say the pipeline method is more realistic than joint approaches. Actually for WS and PT available training data are usually larger than those for DP especially in domain adaptation situations.

In this paper we assume that we want accurate DP results of raw texts in a new target domain and the method is language resource addition to each step because it is the easiest and sometimes most effective method for improving the accuracy. We propose several variations of language resource addition strategies and compare them in DP accuracy.

In the experiments, we use core data of Balanced Corpus of Contemporary Written Japanese (BCCWJ) (Maekawa, 2008) which has six sub-domains (Yahoo!Answers, white paper, Yahoo!Blog, book, magazine, and newspaper). We select white paper and Yahoo!Blog as target domains to investigate language resource addition strategies. The reason is that two domains have the lowest DP accuracies and the causes are different. At WS and PT steps, the possible language resources are word dictionaries and fully annotated corpora. At DP step, the language resource is fully annotated corpora.

In the following sections, we briefly explain WS, PT, DP, and language resources for them. Then we experimentally compare the performances in accuracy of DP from raw texts under various settings. Finally we discuss the strategies from the viewpoints of efficiency and cost.

2. Word Segmentation

WS is the first step, which takes an unsegmented text as the input and segments it into a sequence of words. We adopt one of the state-of-art methods, pointwise method (Neubig and Mori, 2010; Neubig et al., 2011; Mori and Neubig, 2014). This method is capable of utilizing various types of language resources and thus useful in domain adaptation situations.

2.1. Method

The pointwise word segmenter takes a sequence of characters $\boldsymbol{x} = x_1 x_2 \cdots x_n$ as input. Then it makes a classification at every point between two characters (decision point). Each classification is binary indicating whether there is a word boundary (Y) or not (N) as shown in Figure 2. Normally SVM (Fan et al., 2008) is used for classification referring to the features described in Table 1. They are derived from surrounding characters $\boldsymbol{x} = x_{i-m+1} x_{i-m+2} \cdots x_{i+m}$, where m is the window size.
Assuming that a discriminant function is $f(\boldsymbol{x}, b_i)$, the word segmenter calculates the following for each i:

$$\hat{b_i} = \underset{b_i \in \{\mathsf{Y},\mathsf{N}\}}{\textbf{argmax}}\, f(\boldsymbol{x}, b_i).$$

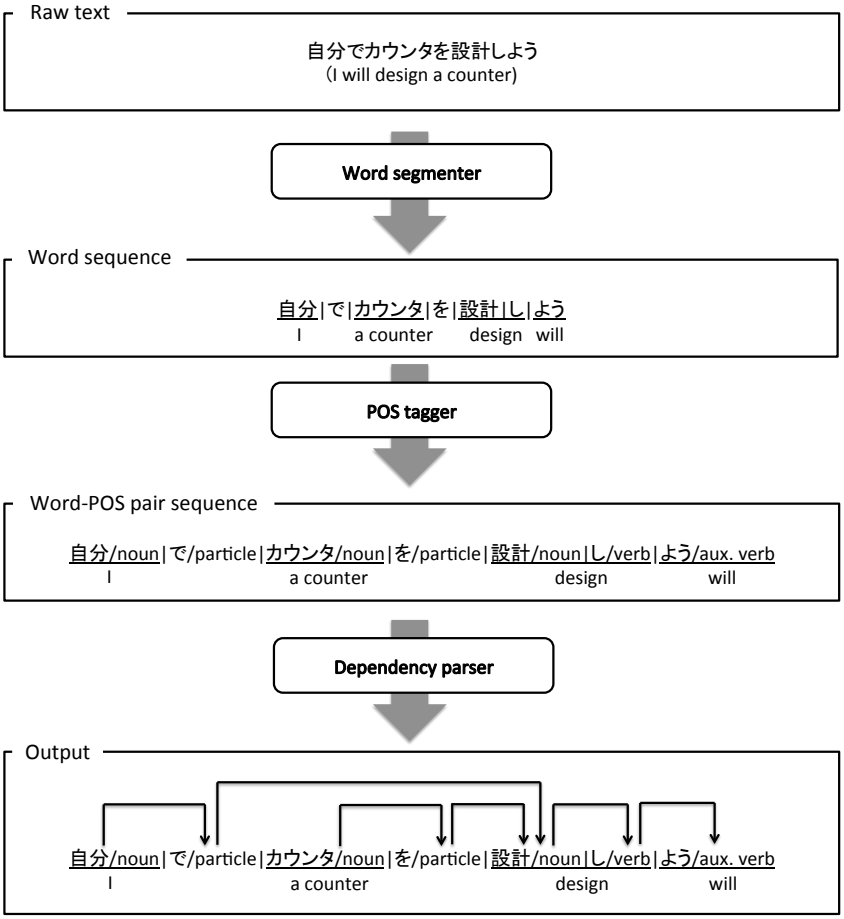

Fig. 1: Pipeline combination of word segmentation, POS tagging, and dependency parsing.

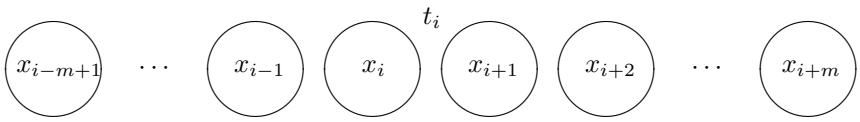

Fig. 2: The characters to be referred to for the decision point t_i.

From the sequence of decisions $\boldsymbol{b} = b_1 b_2 \cdots b_{n-1}$, we obtain a word sequence for the input.

2.2. Language Resources for Word Segmentation

The parameters of $f(\boldsymbol{x}, b_i)$ are estimated from annotated corpora and/or dictionaries (lists of spellings). Below we explain these language resources, with an example sentence "自分でカウンタを設計しよう," where "カウンタ" is a new word.

Word dictionary: We add "カウンタ" to a word dictionary.

Fully annotated corpus: The traditional training data has labels at all decision points as follows.

自分 | で | カウンタ | を | 設計 | し | よう

3. POS Tagging

The second step is PT, which takes a sequence of words as input and estimates a POS tag for each word. For an automatic POS tagger, we adopt pointwise method (Neubig et al., 2011) because of its flexibility of language resource addition.

3.1. Method

The pointwise POS tagger takes a target word w_i with the left character sequence $\boldsymbol{x}_-$ and the right character sequence $\boldsymbol{x}_+$ for the context as shown in Figure 3. Then it estimates the POS tag t_i of w_i by referring to the context. Note that it does not refer to the estimated POS tags of other words nor word boundary information other than w_i to allow flexible language resource addition. Each estimation is multi-class classification and Neubig et al. (2011) reported that SVM (Fan et al., 2008) achieves comparable accuracy to

Table 1: The features for word segmentation.

W1:	character n-gram	We use a sequence of characters around a decision point, whose window size is m and length is n.
W2:	character type n-gram	We translate characters into character types (*kanji*, *katakana*, *hiragana*, Roman letters, numbers, and other), and the feature is a sequence of character types as same as character n-gram.
W3:	word dictionary	We use following flags at the decision point t_i for the length k. (1) whether the left character sequence $x_{1-k+1}x_{1-k+2}\cdots x_i$ is in a dictionary. (2) whether the right character sequence $x_{i+1}x_{1+2}\cdots x_{i+k}$ is in a dictionary. (3) whether the crossing character sequence $x_{i-j+1}x_{i-j+2}\cdots x_{i-j+k}(1 \leq \forall_j < k)$ is in a dictionary.

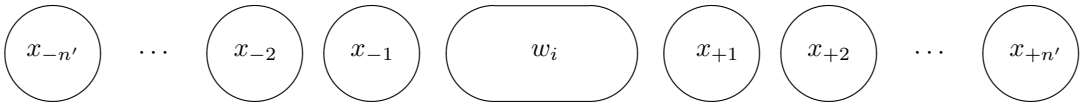

Fig. 3: The characters to be referred to for the word w_i.

Table 2: The features for POS tagging.

P1:	word w_i	
P2:	character n-grams in $\boldsymbol{x}_{-}\boldsymbol{x}_{+}$	A partial sequence of characters whose window size is n' and length is $1, 2, \cdots, n$ around the decision point.
P3:	character type n-grams in $\boldsymbol{x}_{-}\boldsymbol{x}_{+}$	We translate characters into character types. The feature is a part of sequence of character types (same as character n-gram in Table 1).

sequence-based method.

Assuming that a discriminant function is $f(\boldsymbol{w}, t_i)$, the POS tagger calculates the following for each i:

$$\hat{t_i} = \underset{t_i \in T}{\textbf{argmax}}\, f(\boldsymbol{w}, t_i),$$

where T = {noun, verb, particle,aux.verb, ...}. We also use SVM referring to the features described in Table 2.

3.2. Language Resources for POS Tagging

Similar to WS, the pointwise POS tagger is capable of utilizing the following two types of language resources. Here again we use the same example as in WS with a new word "カウンタ/noun."

Word-POS pair dictionary: We add pairs of a word and its POS tag (e.g. "カウンタ/noun") to the dictionary.

Fully annotated corpus: The traditional training data for POS taggers contains a sequence of word-POS pairs. An example is as follows:

```
自分/noun|で/particle|
カウンタ/noun|を/particle|
設計/noun|し/verb|よう/aux.verb.
```

4. Dependency Parsing

DP takes a sequence of words and decides dependency relations among them. Among various solutions, that based on the maximum spanning tree (MST) framework (McDonald and Pereira, 2006; McDonald et al., 2005) is one of the state-of-the-art methods. We adopt an MST variation (Flannery et al., 2012) which is capable of utilizing various types of language resources. Below, we overview the MST dependency parser and explain language resources for it.

4.1. Method

A Dependency parser takes a sequence of words $\boldsymbol{w} = w_1,\ w_2,\ \ldots,\ w_n$ and a sequence of POS tags $\boldsymbol{t} = t_1,\ t_2,\ \ldots,\ t_n$ as input and estimates the dependency tree $\boldsymbol{d} = \langle d_1,\ d_2,\ \ldots,\ d_n \rangle$, where $d_i = j$ indicates that the head of w_i is w_j.

The MST parser firstly assigns edge scores $\sigma(i, j, \boldsymbol{w}) = \frac{\exp(\boldsymbol{\theta} \cdot \boldsymbol{\phi}(i,j,\boldsymbol{w}))}{\sum_j \exp(\boldsymbol{\theta} \cdot \boldsymbol{\phi}(i,j,\boldsymbol{w}))}$, where $\boldsymbol{\phi}(i, j, \boldsymbol{w})$ is a real value vector calculated from the features shown in Table 3 and $\boldsymbol{\theta}$ is a vector of weights for features estimated from a training data. Then the parser searches for the MST as follows:

$$\hat{\boldsymbol{d}} = \underset{\boldsymbol{d} \in D}{\textbf{argmax}} \sum_{i=1}^{n} \sigma(i, j, \boldsymbol{w}).$$

Table 3: The features for dependency parsing.

D1:	The distance between a dependent word w_i and its candidate head w_j.
D2:	The surface forms of w_i and w_j.
D3:	The POSs of w_i and w_j .
D4:	The surface forms of up to three words to the left of w_i and w_j.
D5:	The surface forms of up to three words to the right of w_i and w_j.
D6:	The POSs of the words selected for D4.
D7:	The POSs of the words selected for D5.

Table 4: Corpus specifications.

Usage	Domain	(symbol)	#Sentences	#Words	#Characters	#Words/#Sentences
training	Yahoo!Answers	(OC)	1,614	33,078	46,435	20.49
	white paper	(OW)	1,552	62,735	90,610	40.42
	Yahoo!Blog	(OY)	1,858	31,563	46,481	16.99
	book	(PB)	2,254	53,037	73,194	23.53
	magazine	(PM)	2,514	42,800	65,245	17.02
	newspaper	(PN)	2,590	57,319	83,985	22.13
test	Yahoo!Answers	(OC)	500	9,846	29,038	19.69
	white paper	(OW)	504	23,952	71,352	47.52
	Yahoo!Blog	(OY)	509	9,239	27,208	18.15
	book	(PB)	511	11,792	34,865	23.08
	magazine	(PM)	495	7,415	21,750	14.98
	newspaper	(PN)	505	12,621	37,358	24.99

4.2. Language Resource for Dependency Parsing

For the dependency parser, the language resource is a fully annotated corpus.

Fully Annotated Corpus: Traditionally a training sentence for dependency parsers contains the dependency relationships among all its words. Thus it implies that the sentence is divided into words but it does not need POSs for the words.

5. Strategy Comparison

In order to obtain knowledge of language resource addition strategy, we measured the contributions to the parsing accuracy of language resource additions to the sub-steps. In this section, we first describe the settings of experiments, then evaluate their results, and finally discuss the strategies.

5.1. Language Resources

For the experiments, we used a part of BCCWJ (Maekawa, 2008) annotated with dependency trees provided by (Mori et al., 2014). Thus the sentences are divided into words annotated with POS tags and tree structures. BCCWJ has six sub-domains (Yahoo!Answers, white paper, Yahoo!Blog, book, magazine, and newspaper). We divided each of them into a training and test part (Table 4). We also use a dictionary, UniDic (Den et al., 2007), containing 234,653 word-POS pairs. We separated inflectional endings from the stems.

5.2. Evaluation Criteria

The PT evaluation criterion is the same as morphological analysis F-measure (Nagata, 1994). Let M_{REF} be the number of word-POS pairs in the correct sentences, M_{SYS} be the number of word-POS pairs in the output sentences, and M_{COR} be the number of the word-POS pairs in both of the correct sentences and the output sentences, then the precision P and the recall R are defined as follows:

$$P = M_{COR}/M_{REF},$$

$$R = M_{COR}/M_{SYS}.$$

The total evaluation criterion is F-measure, the harmonic mean of the precision and the recall. For WS evaluation, we only compare words ignoring POSs.

For DP evaluation we count the number of dependency relations whose sources and destinations meet those of the correct relations. A dependency relation has a source word-POS pairs and a destination word-POS pairs. Let D_{REF} be the number of dependency relations in the correct sentences, D_{SYS} be the number of dependency relations in the output sentences, and D_{COR} be the number of the dependency relations in both of the correct sentences and the output sentences, then the precision P and the recall R are defined as follows:

$$P = D_{COR}/D_{REF},$$

$$R = D_{COR}/D_{SYS}.$$

The total evaluation criterion is F-measure, the harmonic mean of precision and recall. Because Japanese dependencies are always from left to right and the last word does not have a destination, we exclude it.

Table 5: The accuracies and coverages of each domain.

Domain (symbol)	Word segmentation (WS)	POS tagging (PT)	Dependency parsing (DP)	Coverage (word)	Coverage (word-POS pair)
Yahoo!Answers (OC)	97.88	95.93	87.09	99.87	99.82
white paper (OW)	98.78	96.67	**82.93**	99.97	99.91
Yahoo!Blog (OY)	**96.52**	93.90	**83.03**	**98.06**	**97.82**
book (PB)	98.05	95.75	84.13	99.95	99.94
magazine (PM)	97.41	94.53	83.75	99.72	99.57
newspaper (PN)	97.64	95.67	83.52	99.37	98.94

Table 6: The accuracies and coverages of white paper (OW).

Language resource addition strategy	Word segmentation (WS)	POS tagging (PT)	Dependency parsing (DP)	Coverage (word)	Coverage (word-POS pair)
`--/--/--`	98.78	96.67	82.93	99.97	99.91
`Wd/--/--`	98.78	96.67	82.93	99.98	99.91
`Wf/--/--`	99.09	96.94	83.30	99.98	99.91
`Wd/Pd/--`	98.78	96.67	82.93	99.98	99.92
`Wf/Pf/--`	99.09	98.25	85.16	99.98	99.94
`Wf/--/Df`	99.09	96.94	84.44	99.98	99.91
`Wf/Pf/Df`	99.09	98.25	86.53	99.98	99.94

Table 7: The accuracies and coverages of Yahoo!Blog (OY).

Language resource addition strategy	Word segmentation (WS)	POS tagging (PT)	Dependency parsing (DP)	Coverage (word)	Coverage (word-POS pair)
`--/--/--`	96.52	93.90	83.03	98.06	97.82
`Wd/--/--`	96.64	94.01	83.16	98.93	97.82
`Wf/--/--`	97.22	94.46	83.64	98.93	97.82
`Wd/Pd/--`	96.64	93.99	83.13	98.93	98.68
`Wf/Pf/--`	97.22	94.99	84.48	98.93	98.77
`Wf/--/Df`	97.22	94.46	84.41	98.93	97.82
`Wf/Pf/Df`	97.22	94.99	85.12	98.93	98.77

We also calculated the coverage of words or word-POS pairs. They are defined as the percentages of the units (tokens) in the test set appearing in the training set or the dictionary.

5.3. Strategies

In this section, we explain language resource addition strategies. We assume frequent adaptation situations where we want accurate dependency parsing results of the texts in a new domain. For example, when we think of new NLP applications, the texts may differ from the general domain texts and we encounter the problem that the accuracy of the default model is very low. Thus in the experimental settings the baseline is the model trained only from a general corpus and dictionary, denoted by `--/--/--` meaning that we use no additional language resources for WS, PT, and DP. The general domain corpus is the concatenation of the sub-domains other than the target domain.

As the language resource addition strategies we add some types of language resources to the baseline as follows to simulate the above-mentioned situation.

1. `--/--/--` **:** The baseline.

2. `Wd/--/--` **:** Word dictionary. We add words appearing only in the training data in the target domain to the dictionary.

3. `Wf/--/--` **:** Full annotation for WS. We add the training corpus in the target domain discarding the POS and dependency information. Thus the corpus contains word sequences.

4. `Wd/Pd/--` **:** Word-POS pair dictionary. We add the words in `Wd/--/--` with the POS tag of the first appearance in the target training corpus. We do not add the word with other POSs. The reason is that it is not realistic because we need to check all the appearances of the word in the (training) corpus to do so. Note that this is the superset of `Wd/--/--` .

5. `Wf/Pf/--` **:** Full annotation for PT. We add the training corpus in the target domain discarding the dependency information. Thus the corpus contains word-POS pair sequences. Note that this is the superset of `Wf/--/--` .

6. `Wf/--/Df` **:** We add the training corpus discarding POS information.

7. `Wf/Pf/Df` **:** We add the training corpus.

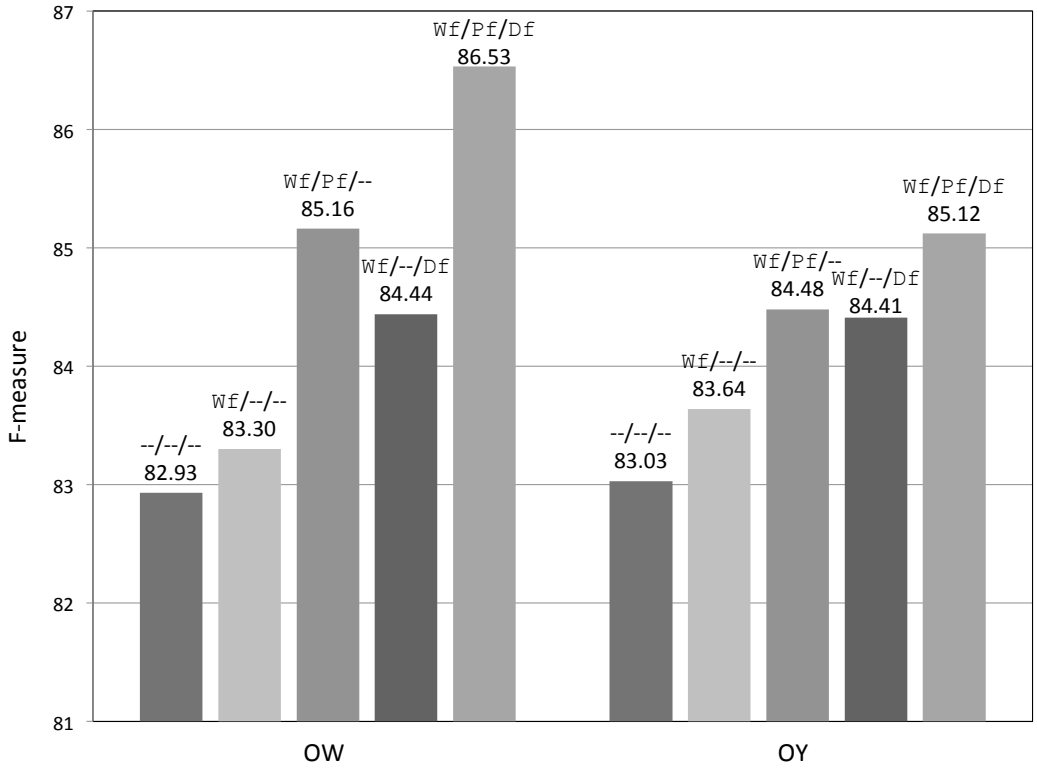

Fig. 4: The accuracies of each domain and strategy.

With these models, we can figure out which annotation is effective for raw text parsing.

5.4. Test Domains

To select the test domain, we conducted parsing experiments without using in-domain data. Table 5 shows the results. From this table, we see that white paper (OW) and Yahoo!Blog (OY) have the lowest DP accuracies. Interestingly the causes are different. For OY the accuracy of the WS step is low and it seems to lower the DP accuracy. On the other hand, the WS and PT accuracies of OW are the highest among six, but the DP accuracy is the lowest. The reason may be that the sentences in OW are written by the professional and very formal, while the sentences in OY are so-called user generated contents and contain many unusual words such as emoticons. The coverages (low for OY and high for OW) shown in Table 5 also support this reasoning. In addition, the sentences of OW tend to be longer than those of others (Table 4), making DP of OW much more difficult. From these observations we selected OW and OY as the target domains to investigate language resource addition strategies.

5.5. Results and Discussion

We conducted the pipeline combination of WS, PT, and DP of the strategies described above and measured the performances at each step. Table 6 and Table 7 show the results. Note that some values are theoretically equal (e.g. the WS accuracies of `Wf/--/--` , `Wf/Pf/--` , `Wf/--/Df` , and `Wf/Pf/Df`).

From the results we see that language resource additions improve DP accuracies in the both domains. And an annotated corpus addition is always better than a dictionary addition.

To compare the contributions of the annotated corpora to the DP accuracies, we show graphs in Figure 4.

The effectiveness of language resource addition to each step differs greatly depending on the domain. For example, the effectiveness of WS for OW is much lower than that for OY as shown in Figure 4. This is because WS of OY is difficult, and language resources improve the coverage of OY.

Therefore, for OW it is not a good strategy to annotate a target corpus with word segmentation only (`Wf/--/--`). Instead it is a better strategy to add word-POS pairs (`Wf/Pf/--`). The accuracy of `Wf/Pf/--` in OW (85.16) is higher than that of `Wf/--/Df` in OW (84.44). In addition, dependency annotation is more costly than POS annotation. Thus the POS annotation is much more effective than dependency annotation of OW.

On the other hand, for OY preparing word segmented sentences (`Wf/--/--`) is not a bad annotation strategy. Because word boundary annotation is the less costly, this strategy improves most effectively when we can spare only small cost for adaptation. The accuracy of `Wf/Pf/--` (84.48) is slightly higher than that of `Wf/--/Df` (84.41) for OY, we can say that POS annotation is more effective considering the annotation costs.

In summary, the effectiveness of language resource addition to each step depends on the domain, or the characteristics of the domain. For well-written texts of limited vocabulary like OW, an effective language resource is a word-POS

pair sequence corpus. Contrary for sentences in user generated contents like OY, adding a corpus containing word sequences or word-POS pair sequences is a good strategy depending on how much you want to improve the accuracy. We may guess which type a new target domain falls into by checking the coverage of the target domain on a small portion of sentences segmented into words. When the target domain has low coverage like user-created content, the strategy in OY is suitable. When we are targeting texts whose coverage is very high, the strategy of OW allows us to improve the accuracy more effectively.

6. Conclusion

In this paper, we surveyed how to improve the accuracy of DP by adding language resources on the premise of pipeline method.

The language resources are word boundary information, POS tags, and dependency relations. We compared various methods for adding language resources. The experimental results indicated that the effectiveness of language resource addition to each step is different depending on the target domain. For example, when we handle well-written texts of limited vocabulary, an effective language resource is a word-POS pair sequence corpus. Thus it is important to check out the characteristics of the target domain and select a language resource addition strategy for an efficient DP accuracy improvement.

7. Acknowledgments

This work was supported by JSPS Grants-in-Aid for Scientific Research Grant Number 26280084 and NTT agreement dated 06/08/2015.

8. Bibliographical References

Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, pages 149–164.

Yasuharu Den, Toshinobu Ogiso, Hideki Ogura, Atsushi Yamada, Nobuaki Minematsu, Kiyotaka Uchimoto, and Hanae Koiso. 2007. The development of an electronic dictionary for morphological analysis and its application to Japanese corpus linguistics. *Japanese Linguistics*, 22:101–122.

Jason M. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th International Conference on Computational Linguistics*, pages 340–345.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

Daniel Flannery, Yusuke Miyao, Graham Neubig, and Shinsuke Mori. 2012. A pointwise approach to training dependency parsers from partially annotated corpora. *Journal of Natural Language Processing*, 19(3).

Jun Hatori, Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. 2012. Incremental joint approach to word segmentation, POS tagging, and dependency parsing in Chinese. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 1045–1053.

Terry Koo and Michael Collins. 2010. Efficient third-order dependency parsers. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1–11.

Taku Kudo and Yuji Matsumoto. 2002. Japanese dependency analysis using cascaded chunking. In *Proceedings of the Sixth Conference on Computational Natural Language Learning*, pages 63–69.

Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. 2004. Applying conditional random fields to Japanese morphological analysis. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 230–237.

Sadao Kurohashi and Makoto Nagao. 1994. KN parser : Japanese dependency/case structure analyzer. In *Proceedings of the International Workshop on Sharable Natural Language Resources*, pages 48–55.

Sadao Kurohashi, Toshihisa Nakamura, Yuji Matsumoto, and Makoto Nagao. 1994. Improvements of Japanese morphological analyzer JUMAN. In *Proceedings of the International Workshop on Sharable Natural Language Resources*, pages 22–28.

Kikuo Maekawa. 2008. Balanced corpus of contemporary written Japanese. In *Proceedings of the 6th Workshop on Asian Language Resources*, pages 101–102.

Yuji Matsumoto, Akira Kitauchi, Tatsuo Yamashita, Yoshitaka Hirano, Hiroshi Matsuda, Kazuma Takaoka, and Masayuki Asahara. 2000. Morphological analysis system ChaSen version 2.2.1 manual. *Nara Institute of Science and Technology*.

Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proceedings of the Eleventh European Chapter of the Association for Computational Linguistics*, pages 81–88.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 523–530.

Ryan McDonald, Kevin Lerman, and Fernando Pereira. 2006. Multilingual dependency analysis with a two-stage discriminative parser. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, pages 216–220.

Shinsuke Mori and Graham Neubig. 2014. Language resource addition: Dictionary or corpus? In *Proceedings of the Nineth International Conference on Language Resources and Evaluation*, pages 1631–1636.

Shinsuke Mori, Masafumi Nishimura, Nobuyasu Itoh, Shiho Ogino, and Hideo Watanabe. 2000. A stochastic parser based on a structural word prediction model. In *Proceedings of the 18th International Conference on Computational Linguistics*, pages 558–564.

Shinsuke Mori, Hideki Ogura, and Tetsuro Sasada. 2014. A Japanese word dependency corpus. In *Proceedings of*

the Nineth International Conference on Language Resources and Evaluation, pages 753–758.

Masaaki Nagata. 1994. A stochastic Japanese morphological analyzer using a forward-DP backward-A* N-Best search algorithm. In *Proceedings of the 15th International Conference on Computational Linguistics*, pages 201–207.

Graham Neubig and Shinsuke Mori. 2010. Word-based partial annotation for efficient corpus construction. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation*, pages 2723–2727.

Graham Neubig, Yosuke Nakata, and Shinsuke Mori. 2011. Pointwise prediction for robust, adaptable Japanese morphological analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 529–533.

Joakim Nivre and Jens Nilsson. 2005. Pseudo-projective dependency parsing. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 99–106.

Joakim Nivre and Mario Scholz. 2004. Deterministic dependency parsing of English text. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 64–70.

Kemal Oflazer. 1999. Dependency parsing with an extended finite state approach. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 254–260.

Kenji Sagae and Alon Lavie. 2006. A best-first probabilistic shift-reduce parser. In *Proceedings of the 21st International Conference on Computational Linguistics*.