Japanese Dependency Parsing Using Sequential Labeling for Semi-spoken Language

Kenji Imamura and **Genichiro Kikui**
NTT Cyber Space Laboratories, NTT Corporation
1-1 Hikarinooka, Yokosuka-shi, Kanagawa, 239-0847, Japan
{imamura.kenji, kikui.genichiro}@lab.ntt.co.jp

Norihito Yasuda
NTT Communication Science Laboratories, NTT Corporation
2-4 Hikaridai, Seika-cho, Soraku-gun, Kyoto, 619-0237, Japan
n-yasuda@cslab.kecl.ntt.co.jp

Abstract

The amount of documents directly published by end users is increasing along with the growth of Web 2.0. Such documents often contain spoken-style expressions, which are difficult to analyze using conventional parsers. This paper presents dependency parsing whose goal is to analyze Japanese semi-spoken expressions. One characteristic of our method is that it can parse self-dependent (independent) segments using sequential labeling.

1 Introduction

Dependency parsing is a way of structurally analyzing a sentence from the viewpoint of modification. In Japanese, relationships of modification between phrasal units called *bunsetsu* segments are analyzed. A number of studies have focused on parsing of Japanese as well as of other languages. Popular parsers are CaboCha (Kudo and Matsumoto, 2002) and KNP (Kurohashi and Nagao, 1994), which were developed to analyze formal written language expressions such as that in newspaper articles.

Generally, the syntactic structure of a sentence is represented as a tree, and parsing is carried out by maximizing the likelihood of the tree (Charniak, 2000; Uchimoto et al., 1999). Units that do not modify any other units, such as fillers, are difficult to place in the tree structure. Conventional parsers have forced such independent units to modify other units.

Documents published by end users (e.g., blogs) are increasing on the Internet along with the growth of Web 2.0. Such documents do not use controlled written language and contain fillers and emoticons. This implies that analyzing such documents is difficult for conventional parsers.

This paper presents a new method of Japanese dependency parsing that utilizes sequential labeling based on conditional random fields (CRFs) in order to analyze semi-spoken language. Concretely, sequential labeling assigns each segment a dependency label that indicates its relative position of dependency. If the label set includes self-dependency, the fillers and emoticons would be analyzed as segments depending on themselves. Therefore, since it is not necessary for the parsing result to be a tree, our method is suitable for semi-spoken language.

2 Methods

Japanese dependency parsing for written language is based on the following principles. Our method relaxes the first principle to allow self-dependent segments (c.f. Section 2.3).

1. Dependency moves from left to right.
2. Dependencies do not cross each other.
3. Each segment, except for the top of the parsed tree, modifies at most one other segment.

2.1 Dependency Parsing Using Cascaded Chunking (CaboCha)

Our method is based on the cascaded chunking method (Kudo and Matsumoto, 2002) proposed as the CaboCha parser [1]. CaboCha is a sort of shift-reduce parser and determines whether or not a segment depends on the next segment by using an

[1] http://www.chasen.org/˜taku/software/cabocha/

Proceedings of the ACL 2007 Demo and Poster Sessions, pages 225–228,
Prague, June 2007.

SVM-based classifier. To analyze long-distance dependencies, CaboCha shortens the sentence by removing segments for which dependencies are already determined and which no other segments depend on. CaboCha constructs a tree structure by repeating the above process.

2.2 Sequential Labeling

Sequential labeling is a process that assigns each unit of an input sequence an appropriate label (or tag). In natural language processing, it is applied to, for example, English part-of-speech tagging and named entity recognition. Hidden Markov models or conditional random fields (Lafferty et al., 2001) are used for labeling. In this paper, we use linear-chain CRFs.

In sequential labeling, training data developers can design labels with no restrictions.

2.3 Cascaded Chunking Using Sequential Labeling

The method proposed in this paper is a generalization of CaboCha. Our method considers not only the next segment, but also the following N segments to determine dependencies. This area, including the considered segment, is called the *window*, and N is called the window size. The parser assigns each segment a dependency label that indicates where the segment depends on the segments in the window. The flow is summarized as follows:

1. Extract features from segments such as the part-of-speech of the headword in a segment (c.f. Section 3.1).
2. Carry out sequential labeling using the above features.
3. Determine the actual dependency by interpreting the labels.
4. Shorten the sentence by deleting segments for which the dependency is already determined and that other segments have never depended on.
5. If only one segment remains, then finish the process. If not, return to Step 1.

An example of dependency parsing for written language is shown in Figure 1 (a).

In Steps 1 and 2, dependency labels are supplied to each segment in a way similar to that used by other sequential labeling methods. However, our sequential labeling has the following characteristics since this task is dependency parsing.

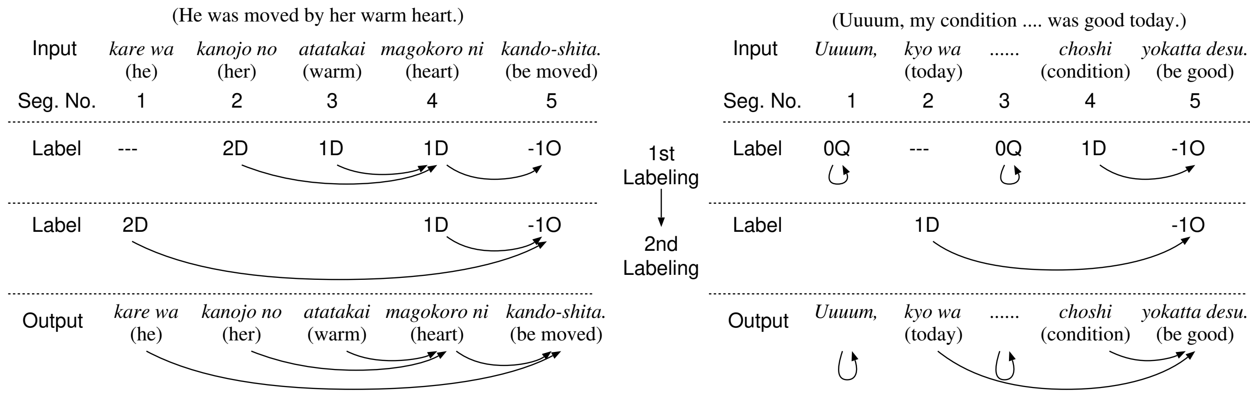
Label	Description
—	Segment depends on a segment outside of window.
0Q	Self-dependency
1D	Segment depends on next segment.
2D	Segment depends on segment after next.
-1O	Segment is top of parsed tree.

Table 1: Label List Used by Sequential Labeling (Window Size: 2)

- The labels indicate relative positions of the dependent segment from the current segment (Table 1). Therefore, the number of labels changes according to the window size. Long-distance dependencies can be parsed by one labeling process if we set a large window size. However, growth of label variety causes data sparseness problems.
- One possible label is that of self-dependency (noted as '0Q' in this paper). This is assigned to independent segments in a tree.
- Also possible are two special labels. Label '-1O' denotes a segment that is the top of the parsed tree. Label '—' denotes a segment that depends on a segment outside of the window. When the window size is two, the segment depends on a segment that is over two segments ahead.
- The label for the current segment is determined based on all features in the window and on the label of the previous segment.

In Step 4, segments, which no other segments depend on, are removed in a way similar to that used by CaboCha. The principle that dependencies do not cross each other is applied in this step. For example, if a segment depends on a segment after the next, the next segment cannot be modified by other segments. Therefore, it can be removed. Similarly, since the '—' label indicates that the segment depends on a segment after N segments, all intermediate segments can be removed if they do not have '—' labels.

The sentence is shortened by iteration of the above steps. The parsing finishes when only one segment remains in the sentence (this is the segment



(a) Written Language

(b) Semi-spoken Language

Figure 1: Examples of Dependency Parsing (Window Size: 2)

Corpus	Type	# of Sentences	# of Segments
Kyoto	Training	24,283	234,685
	Test	9,284	89,874
Blog	Training	18,163	106,177
	Test	8,950	53,228

Table 2: Corpus Size

at the top of the parsed tree). In the example in Figure 1 (a), the process finishes in two iterations.

In a sentence containing fillers, the self-dependency labels are assigned by sequential labeling, as shown in Figure 1 (b), and are parsed as independent segments. Therefore, our method is suitable for parsing semi-spoken language that contains independent segments.

3 Experiments

3.1 Experimental Settings

Corpora In our experiments, we used two corpora. One is the Kyoto Text Corpus 4.0 [2], which is a collection of newspaper articles with segment and dependency annotations. The other is a blog corpus, which is a collection of blog articles taken as semi-spoken language. The blog corpus is manually annotated in a way similar to that used for the Kyoto text corpus. The sizes of the corpora are shown in Table 2.

Training We used CRF++ [3], a linear-chain CRF training tool, with eleven features per segment. All of these are static features (proper to each segment) such as surface forms, parts-of-speech, inflections of a content headword and a functional headword in a segment. These are parts of a feature set that many papers have referenced (Uchimoto et al., 1999; Kudo and Matsumoto, 2002).

Evaluation Metrics Dependency accuracy and sentence accuracy were used as evaluation metrics. Sentence accuracy is the proportion of total sentences in which all dependencies in the sentence are accurately labeled. In Japanese, the last segment of most sentences is the top of the parsed trees, and many papers exclude this last segment from the accuracy calculation. We, in contrast, include the last one because some of the last segments are self-dependent.

3.2 Accuracy of Dependency Parsing

Dependency parsing was carried out by combining training and test corpora. We used a window size of three. We also used CaboCha as a reference for the set of sentences trained only with the Kyoto corpus because it is designed for written language. The results are shown in Table 3.

CaboCha had better accuracies for the Kyoto test corpus. One reason might be that our method manually combined features and used parts of combinations, while CaboCha automatically finds the best combinations by using second-order polynomial kernels.

For the blog test corpus, the proposed method using the Kyoto+Blog model had the best depen-

[2] http://nlp.kuee.kyoto-u.ac.jp/nl-resource/corpus.html

[3] http://www.chasen.org/˜taku/software/CRF++/

Test Corpus	Method	Training Corpus (Model)	Dependency Accuracy	Sentence Accuracy
Kyoto (Written Language)	Proposed Method (Window Size: 3)	Kyoto	89.87% (80766 / 89874)	48.12% (4467 / 9284)
		Kyoto + Blog	89.76% (80670 / 89874)	47.63% (4422 / 9284)
	CaboCha	Kyoto	**92.03**% (82714 / 89874)	**55.36**% (5140 / 9284)
Blog (Semi-spoken Language)	Proposed Method (Window Size: 3)	Kyoto	77.19% (41083 / 53226)	41.41% (3706 / 8950)
		Kyoto + Blog	**84.59**% (45022 / 53226)	**52.72**% (4718 / 8950)
	CaboCha	Kyoto	77.44% (41220 / 53226)	43.45% (3889 / 8950)

Table 3: Dependency and Sentence Accuracies among Methods/Corpora

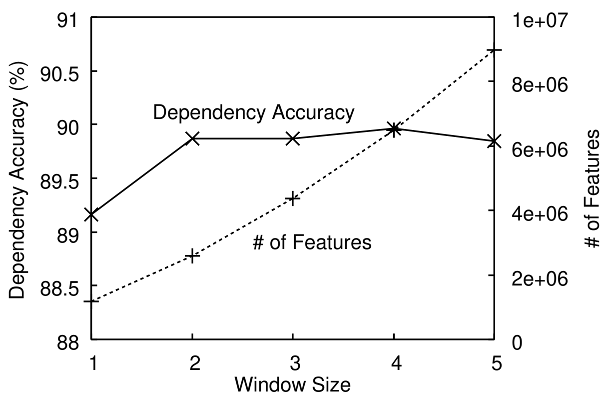


Figure 2: Dependency Accuracy and Number of Features According to Window Size (The Kyoto Text Corpus was used for training and testing.)

dency accuracy result at 84.59%. This result was influenced not only by the training corpus that contains the blog corpus but also by the effect of self-dependent segments. The blog test corpus contains 3,089 self-dependent segments, and 2,326 of them (75.30%) were accurately parsed. This represents a dependency accuracy improvement of over 60% compared with the Kyoto model.

Our method is effective in parsing blogs because fillers and emoticons can be parsed as self-dependent segments.

3.3 Accuracy According to Window Size

Another characteristic of our method is that all dependencies, including long-distance ones, can be parsed by one labeling process if the window covers the entire sentence. To analyze this characteristic, we evaluated dependency accuracies in various window sizes. The results are shown in Figure 2.

The number of features used for labeling increases exponentially as window size increases. However, dependency accuracy was saturated after a window size of two, and the best accuracy was when the window size was four. This phenomenon implies a data sparseness problem.

4 Conclusion

We presented a new dependency parsing method using sequential labeling for the semi-spoken language that frequently appears in Web documents. Sequential labeling can supply segments with flexible labels, so our method can parse independent words as self-dependent segments. This characteristic affects robust parsing when sentences contain fillers and emoticons.

The other characteristics of our method are using CRFs and that long dependencies are parsed in one labeling process. SVM-based parsers that have the same characteristics can be constructed if we introduce multi-class classifiers. Further comparisons with SVM-based parsers are future work.

References

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proc. of NAACL-2000*, pages 132–139.

Taku Kudo and Yuji Matsumoto. 2002. Japanese dependency analyisis using cascaded chunking. In *Proc. of CoNLL-2002*, Taipei.

Sadao Kurohashi and Makoto Nagao. 1994. A syntactic analysis method of long Japanese sentences based on the detection of conjunctive structures. *Computational Linguistics*, 20(4):507–534.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of ICML-2001*, pages 282–289.

Kiyotaka Uchimoto, Satoshi Sekine, and Hitoshi Isahara. 1999. Japanese dependency structure analysis based on maximum entropy models. In *Proc. of EACL'99*, pages 196–203, Bergen, Norway.