A Fast and Accurate Partially Deterministic Morphological Analysis

Hajime Morita **Tomoya Iwakura**
Fujitsu Laboratories Ltd., Kanagawa, Japan
{hmorita, iwakura.tomoya}@fujitsu.com

Abstract

This paper proposes a partially deterministic morphological analysis method for improved processing speed. Maximum matching is a fast deterministic method for morphological analysis. However, the method tends to decrease performance due to lack of consideration of contextual information. In order to use maximum matching safely, we propose the use of Context Independent Strings (CISs), which are strings that do not have ambiguity in terms of morphological analysis. Our method first identifies CISs in a sentence using maximum matching without contextual information, then analyzes the unprocessed part of the sentence using a bi-gram-based morphological analysis model. We evaluate the method on a Japanese morphological analysis task. The experimental results show a 30% reduction of running time while maintaining improved accuracy.

1 Introduction

Morphological analysis segments a text into a sequence of morphemes. The analysis is the first step of analyzing languages such as Chinese, Thai and Japanese. In Japanese, the analysis is a compound task of word segmentation, POS tagging, recognition of conjugation and lemmatization.

Japanese morphological analysis derives a word dictionary. The dictionary is a set of tuples of word surface (conjugated form), base form (lemma of words) and POS tags. Given an input string of a sentence, the morphological analyzer enumerates all substrings of the sentence that are listed in the dictionary. Then the analyzer builds a large lattice (DAG) of the words as a set of candidate word sequences. Because the lattice often becomes large, the lattice building tends to be a bottle neck of the morphological analysis. Finally, the analyzer selects a path on the lattice and gets the analysis result.

Morphological analysis is often used as a preprocessing step for shallow analyses on a large amount of texts collected from Web. These analyses include word counting, keyword extraction and named entity recognition. Morphological analysis tends to occupy a larger part of the processing time in the tasks. Therefore, we investigated a new morphological analysis method that improves efficiency and performance.

There are three kinds of Japanese morphological analysis methods in terms of the range of using contexts. The first group is maximum matching-based methods that do not use context (Sproat et al., 1996; Nagata, 1997; Sassano, 2014). Maximum matching is a significantly fast deterministic method for morphological analysis. However, this method tends to have a low performance, because it often fails an segmentation of phrases which have ambiguity depending on their context. The second group is neural network based methods that do not have explicit limit to the range of context (Morita et al., 2015; Chen et al., 2015; Wang et al., 2016; Kurita et al., 2017; Tolmachev et al., 2018). However, neural network based methods generally require heavy computational cost. The third group is commonly used local context based methods (Kudo et al., 2004; Neubig et al., 2011; Zheng et al., 2013; Kaji and Kitsuregawa, 2013; Kurohashi and Kawahara, 2014). The methods typically use bi-grams as their context. Among these methods, the bi-gram based methods are faster than neural network based methods and their performance is not far from that of neural network based methods.

Proceedings of Recent Advances in Natural Language Processing, pages 804–809,
Varna, Bulgaria, Sep 2–4, 2019.
https://doi.org/10.26615/978-954-452-056-4_093

This paper proposes a partially deterministic morphological analysis method for improved processing speed. We improve processing speed by incorporating maximum matching (the first group) to a bi-gram-based morphological engine (the third group).

In order to alleviate the performance loss of maximum matching, we propose a concept of Context Independent Strings (CISs), which are strings having no ambiguity in terms of morphological analysis. We also propose an algorithm for the building of the CISs dictionary from the large amount of automatically analysed texts. The dictionary maps CISs to the results of morphological analysis (sequence of words and POS tags).

Our method first applies maximum matching to a sentence using the CIS dictionary. CISs in a sentence are deterministically analyzed by the dictionary. Then the method analyze the rest part of the sentence using a bi-gram-based morphological analysis. We can use high-performance and computationally heavy methods such as neural network based methods for the building of the dictionary. That is, we can partially use their analysis results without recomputing through the CIS dictionary.

We obtain CISs from automatically parsed large texts with a state-of-the-art morphological analyzer JUMAN++ (Morita et al., 2015; Tolmachev et al., 2018). We evaluate this method on a Japanese morphological analysis task. The experimental results show a 30% reduction of running time while maintaining improved accuracy.

2 Context Independent Strings

The key idea of using maximum matching without performance loss is to separate character strings into two classes: context independent string (CIS) and context dependent string (CDS).

A CIS is a string that has a one-to-one correspondence with its grammatical analysis result, while a CDS has two or more grammatical analysis results that depend on its contexts. A CIS has to satisfy two conditions:

- The string does not have two or more grammatical analysis results.
- The analysis result is not affected by any strings adjacent to the beginning and end of the strings.

We describe CISs with the following three examples.

(A) 床の間
(alcove, or gap of floor)
(B) 南京
(Nanjing)
(C) 南 京都 病院
(South) (Kyoto) (hopital)

(A) is a string that does not satisfy the first condition of CIS. “床の間” has two grammatical analysis result: “床の間 (alcove)” with one word or “床 (floor) + の (to) + 間 (gap)” with three words. The analysis depends on the context where the string occurred.

In (B), the string “南京” has only one grammatical analysis result “南京” (Nanjin). However, the string is a part of (C) “南京都病院”, and the string corresponts “南” (South) + the first character of “京都” (Kyoto). That violates the second condition of CIS.

When we segment the example (C) “南京都病院” (South Kyoto Hospital) using word-level maximum matching, the example is segmented into “南京+都+病院” (Nanjin metropolitan hospital) instead of correct segmentation “南+京都+病院” (South Kyoto hospital). This is because the longest word matched from the beginning of the example (C) is CDS (“南京”).

In fact, whole (C) is an example of a CIS. There is no “南京+都+病院” (Nanjin metropolitan hospital)“, and then the string has only one grammatical analysis result.

3 Proposed Method

This section describes the building method of a CIS dictionary, and deterministic morphological analysis using the CIS dictionary.

3.1 Building a CISs Dictionary

A CIS dictionary is a data structure where each CIS is associated to its morphological analysis result.

We built the CIS dictionary using the algorithm showed in **Algorithm 1**. In this method, the dictionary is built from a large automatically analyzed corpus C. We collected word N-grams n_j, their surfaces $\mathrm{surface}(n_j)$ and pointers j of the sentences in which the N-grams occurred, as a set of candidates (H, N, S).

In STEP (1) at the line 14 of Algorithm 1, when an identical surface was associated with two or

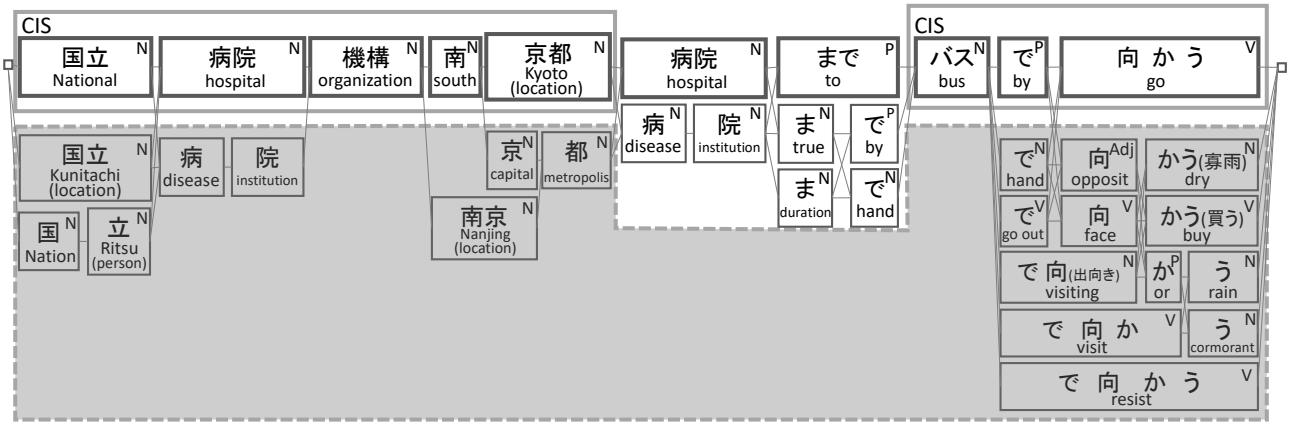

Figure 1: Example of a lattice that is built by our baseline method and our proposed model using a CIS dictionary. The CIS dictionary contains “国立病院機構南京都” and “バスで向かう”, so the corresponding results are loaded from the dictionary. The word candidates surrounded by dotted line are not generated in our proposed model. Our implementation builds a CIS dictionary from word sequences of length 5 or less. Thus, even though “国立病院機構南京都病院” (6 words) is clearly a CIS, it is not in the dictionary.

more N-grams, or the N-grams occurred less frequently than a predetermined threshold T, the surfaces are discarded from the candidates. We aim to discard ambiguous strings similar to example (A).

In STEP (2) at the line 15, we remove CDSs that violate the second condition of CIS. As we can see from Example (B), the surface of CDS appears in a part of other word N-gram. The $\mathrm{findString}$ gives a set of sentences where the string p occurs in corpus C. The step checks there is no occurrence of the string p that appears in a part of another N-gram. We discarded any surface that occurred in a sentence in which the corresponding N-gram did not occur. This is because the occurrence indicates that the surface was a part of another different N-gram.

In STEP (3) at the line 15, the step is filtering using a heuristic. If the corpus is sufficiently large, the process considers any string before and after the N-gram. However, it is infeasible to cover all contexts of all expressions in a corpus of limited size. For the purpose of augmenting the corpus, we discarded the N-grams where words at the beginning or ending of N-gram are sensitive to preceding and succeeding strings. Specifically, if the beginning or end of the N-gram is a one-character word and is not in a predefined white list (punctuations, and a part of particles), the N-gram is discarded by this step. We assume that one-character words are often a suffix or prefix of other words and these words are sensitive to the surrounding contexts. We show the process as $\mathrm{isSensitive}$ function in Algorithm 1.

Finally, we build a CIS dictionary D using the collected CISs and their analysis result.

3.2 Analysis using a CIS Dictionary

There are three steps to analyse a sentence using a CIS dictionary: (1) Deterministic analysis by maximum matching with a CIS dictionary, (2) Building lattice by lookup dictionary and (3) Search for the least cost path on the lattice. We show our algorithm in Algorithm 2.

First, **LookUpLongestCIS** in Algorithm 2 finds CISs with the maximum match of a CIS dictionary from each character of the input sentence. If a CIS is found, the algorithm skips the range of matched string, and calls **LookUpLongestCIS** again. If it is not found, the step is restarted from the next character. The algorithm calls *LookUpDictionary* and builds a word lattice for each span where any analysis result is not obtained with the maximum matching.

Finally, whole sentence is connected as one lattice, and a word sequence with the lowest cost is obtained by Viterbi algorithm. On this lattice, the spans deterministically analyzed are expressed as nodes corresponding to the analysis result obtained from the dictionary, and their costs are 0. Other costs are calculated by a bi-gram model.

Figure 1 shows an example of a lattice that is built by our proposed method.

Algorithm 1 An algorithm for building a CIS dictionary

T is a threshold for N-gram frequency,
$C \leftarrow \{s_0, \ldots, s_M\}$,
$S \leftarrow \{\}$
H, R, D are associative arrays. H maps surface to array of sentence ids, R maps surface to array of analysis result, D maps surface to an analysis result
for $i = 0$ to N **do**
  for all $n_j \in s_i$ **do**
    insert$(H[\text{surface}(n_j)], i)$
    insert$(R[\text{surface}(n_j)], n_j)$
    insert$(S, \text{surface}(n_j))$
  end for
end for
for all $p \in S$ **do**
  # STEP (1)
  next if $|R[p]| > 1$ **OR** $|H[p]| < T$
  # STEP (2)
  next if $H[p] \neq \text{findString}(p, C)$
  # STEP (3)
  next if isSensitive$(R[p][0])$
  $D[p] \leftarrow (R[p][0])$
end for
return D

Algorithm 2 Morphological analysis using a CIS dictionary and Regular Expressions

S $\leftarrow \{c_0, \ldots, c_n\}$, $i \leftarrow 0$, $j \leftarrow 0$, $k \leftarrow 0$
L = A lattice structure
while $i \leq N$ **do**
  result, length $\leftarrow$ **LookUpLongestCIS**(S, i)
  if result != ϕ **then**
    L[i] $\leftarrow$ result
    $i \leftarrow i$ + length
  else
    $i \leftarrow i + 1$
  end if
end while
for $j = 0$ to N –(2) **do**
  if j is processed by **LookUpLongestCIS** **then**
    continue
  end if
  L[j] $\leftarrow$ *LookUpDictionary*(S,j)
end for
return Viterbi(L) –(3)

4 Experiments

4.1 Data and Models

We build a CIS dictionary using Mainichi News articles published from 1991 to 2010. The corpus contains approximately 2 million articles. We analyzed the corpus using JUMAN++ v1.02 (Morita et al., 2015). We set threshold T to 4, and limited the maximum N-gram length to 5.

A morphological dictionary used in our models is stemmed from JUMAN++ v1.02. We trained our feature parameters of CRF using L-BFGS (Kudo et al., 2004; Liu and Nocedal, 1989). We trained our models using Kyoto University Web Document Leads Corpus (Hangyo et al., 2012; Kawahara et al., 2014) that contains approximately 15,000 manually annotated sentences. The corpus contains manually annotated word segmentations, POS tags, dependencies, predicate-argument structures including zero anaphora, and so on. We used sentences with id (w201106-00000–w201106-00023) for training and sentences with id (w201106-00024) for development.

We evaluated the performance of our methods by KNB corpus (Hashimoto et al., 2011) that consists of 249 articles (4,186 sentences). The corpus contains manually annotatted word segmentation, POS tags and dependencies. Then, we evaluated the efficiency of our models by measuring running times of analysis on news articles of Yomiuri Shimbun in 2013. The text consists of approximately 300 thousand articles (approximately 4 million sentences).

We measured the performance of the methods by two performance measures Word and Word+POS. Word is the F-value of word segmentation, and Word+POS is the F-value of joint evaluation of word segmentation and POS tagging.

4.2 Performance Evaluation

We compared our proposed model with the baseline and JUMAN++. Partially deterministic analysis using a CIS dictionary did not lose to their performance but slightly improved in both Word and Word+POS. Our baseline is almost a reimplementation of MeCab (Kudo et al., 2004) and performs almost equally. JUMAN++ is an upper bound of our proposed model because a deterministically analyzed part of the analysis result is almost equivalent with the result of JUMAN++.

Table 1: Performance comparison of various systems.

Method	Word (F-val)	Word+POS (F-val)
Baseline	95.94	95.07
Partially deterministic (Proposed)	**95.99**	**95.17**
JUMAN++	96.84	96.15

Table 2: Comparing running times of various systems.

Method	Time (seconds)
Baseline	315.3
Partially deterministic (Proposed)	**223.4**
MeCab	124.5
JUMAN++	341705.7

4.3 Running Time Comparison

We compared our systems with publicly available implementations. MeCab is the fastest, but our implementation (Baseline) marks similar running time. Since our baseline model is not largely different from MeCab, the difference of running time is due to the implementation. Partially deterministic analysis reduces running time by 30%. If we implement our proposed method on MeCab or JUMAN++, we suspect it will improve their efficiency.

5 Conclusion

We presented a new fast partially deterministic morphological analysis method. We introduced the concept of Context Independent Strings and presented an algorithm for building a dictionary of CISs from a large corpus. We checked that the proposed method improved both efficiency and performance of morphological analysis. The method reduced the running time by 30% and improved the performance 0.1 pt in Word+POS.

References

Xinchi Chen, Xipeng Qiu, Chenxi Zhu, Pengfei Liu, and Xuanjing Huang. 2015. Long short-term memory neural networks for chinese word segmentation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. pages 1197–1206.

Masatsugu Hangyo, Daisuke Kawahara, and Sadao Kurohashi. 2012. Building a diverse document leads corpus annotated with semantic relations. In *Proceedings of the 26th Pacific Asia Conference on Language, Information, and Computation*. Faculty of Computer Science, Universitas Indonesia, Bali,Indonesia, pages 535–544.

Chikara Hashimoto, Sadao Kurohashi, Keiji Shinzato, and Nagata Masaaki. 2011. Construction of a blog corpus with syntactic, anaphoric, and sentiment annotations (in japanese). *Journal of Natural Language Processing* 18(2):175–201.

Nobuhiro Kaji and Masaru Kitsuregawa. 2013. Efficient word lattice generation for joint word segmentation and pos tagging in japanese. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*. pages 153–161.

Daisuke Kawahara, Yuichiro Machida, Tomohide Shibata, Sadao Kurohashi, Hayato Kobayashi, and Manabu Sassano. 2014. Rapid development of a corpus with discourse annotations using two-stage crowdsourcing. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. Dublin City University and Association for Computational Linguistics, Dublin, Ireland, pages 269–278.

Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. 2004. Applying conditional random fields to japanese morphological analysis. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*. Association for Computational Linguistics, Barcelona, Spain, pages 230–237.

Shuhei Kurita, Daisuke Kawahara, and Sadao Kurohashi. 2017. Neural joint model for transition-based chinese syntactic analysis. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL2017)*. Association for Computational Linguistics.

Sadao Kurohashi and Daisuke Kawahara. 2014. Juman ver.7.01. http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?JUMAN.

D. C. Liu and J. Nocedal. 1989. On the limited memory bfgs method for large scale optimization. *Math. Program.* 45(3):503–528.

Hajime Morita, Daisuke Kawahara, and Sadao Kurohashi. 2015. Morphological analysis for unsegmented languages using recurrent neural network language model. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. pages 2292–2297.

Masaaki Nagata. 1997. A self-organizing japanese word segmenter using heuristic word identification and re-estimation. In *Proceedings of the 5th Workshop on Very Large Corpora*. pages 203–215.

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: Human Language Technologies: Short Papers - Volume 2*. HLT '11, pages 529–533.

Manabu Sassano. 2014. Deterministic word segmentation using maximum matching with fully lexicalized rules. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, volume 2: Short Papers*. pages 79–83.

Richard Sproat, William Gale, Chilin Shih, and Nancy Chang. 1996. A stochastic finite-state word-segmentation algorithm for chinese. *Comput. Linguist.* 22(3):377–404.

Arseny Tolmachev, Daisuke Kawahara, and Sadao Kurohashi. 2018. Juman++: A morphological analysis toolkit for scriptio continua. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, Brussels, Belgium, pages 54–59. https://doi.org/10.18653/v1/D18-2010.

Heng-Jun Wang, Nian-Wen Si, and Cheng Chen. 2016. An effective joint model for chinese word segmentation and pos tagging. In *Proceedings of the 2016 International Conference on Intelligent Information Processing*. ICIIP '16, pages 35:1–35:6.

Xiaoqing Zheng, Hanyang Chen, and Tianyu Xu. 2013. Deep learning for Chinese word segmentation and POS tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. pages 647–657.