E-006

Using Feature Selection with Support Vector Machines for Japanese Word Segmentation

Tadashi Yanagihara[†] Kazushi Ikeda[†] Gen Hattori[†] Kazunori Matsumoto[†] Yasuhiro Takishima[†]

1. Introduction

Word segmentation in natural language processing is a task of splitting a given character string into its component words. With languages based on Latin characters, the word segmentation is trivial because a space character is given to be a word delimiter. However, the word delimiter is not trivial with many Asian languages (e.g. Chinese, Japanese, Korean, Vietnamese), and must be estimated. This has led to a long investigation on how to estimate word segmentations in languages mainly found in the Asian region.

Considering Japanese, much research has been conducted in this field, making many morphological analyzers[1][2][3] available to public. All of these methods use a dictionary, which is a list of registered words with values assigned to represent its cost. The benefits of using a dictionary is that it is fast and efficient. The drawback is that the cost of constructing and updating the dictionary is expensive.

Previous work[4] have proposed a scheme for word segmentation based on *n*-gram models. An *n*-gram model used in word segmentation is based on the co-occurrence of two adjacent character strings. In comparison with using a dictionary, the *n*-gram model approach is more easier to construct and modify, because the co-occurrence of the two characters can be easily calculated by a computer.

The method proposed in [4] uses a 2x2 contingency table to determine whether a pair of strings are related to each other. Their method use a scoring scheme[5] based on AIC[6] to examine the relationship of a pair of strings. If the appearance of a given pair of strings are found to be related to each other, we can estimate that they are more likely to form a word. However, the method does not consider the presence of a word segmentation label when creating the contingency table to calculate the score assigned to a word. In some cases, it is possible the score can disagree with the co-occurance of a pair of strings, therefore can have a negative effect towards accuracy. In this paper, we extend the method mentioned in [4] by using a 2x4 contingency table instead of a 2x2 contingency table in order to also consider of the presence of the word segmentation.

2. Problem Formulation

In this section, we illustrate the problem of estimating a word segmentation using an *n*-gram model. First, a given character string can be represented as S. S consists of k characters, where each character can be represented as $c_1, c_2, \ldots c_k$. The area between each character can be represented as candidates for a word segmenation, or boundary, therefore can be represented as b. When the word consists of k characters, the word segmentation can be represented as $b_1, b_2, \ldots b_{k-1}$. The task in word segmentation is guessing the value of b on whether it is a word segmentation (+1) or not (-1).

As an example, we show in Figure 1 the process of determining whether b_3 is a word segmentation in the 6 letter word "FOOBAR". Using features surrounding b_3 can help determine the value of b. One suggestion is using the characters immediately after b_3, (e.g. c_4, which is represented as c) or using combinations of adjacent characters surrounding b_3 (e.g. $c_1 + c_2 + c_3$, which is represented as s). Given these features, we must provide a method to distinguish which features are more useful than the others.

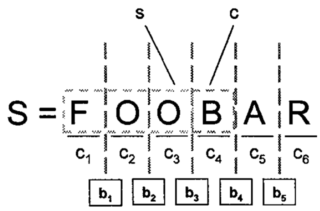

Figure 1: Formulation of the word segmentation problem

3. Proposal

Our process requires two steps; First, calculate the score E to assign towards a given s, c, and b, then transform our features to use as training data for a classifier.

3.1 Calculation Procedure

In [4], four features were used to construct a 2x2 contingency table. Specifically, the four features were: the frequency of when s appeared before c, the frequency of when s appeared before characters other than c, the frequency of when string characters other than s appeared before c, and the frequency of when string characters other than s appeared before characters other than c. We extend the idea of using a 2x2 contingency table to using a 2x4 contingency table, as shown in table1, which now includes information whether b is a word segmenation or not. The necessary values used in a 2x4 contingency table is as follows:

- n_{11} : The frequency of when s is followed by c, and b is a word segmentation.
- n_{12} : The frequency of when s is followed by $\neg c$, and b is a word segmentation.
- n_{13} : The frequency of when $\neg s$ is followed by c, and b is a word segmentation.

[†]KDDI R&D Incorporated

Table 1: 2x4 Contingency Table for s, c and b

	$s \wedge c$	$s \wedge \neg c$	$\neg s \wedge c$	$\neg s \wedge \neg c$
b	n_{11}	n_{12}	n_{13}	n_{14}
$\neg b$	n_{21}	n_{22}	n_{23}	n_{24}

Table 2: Contingency Table for Case M_1

	$s \wedge c$	$s \wedge \neg c$	$\neg s \wedge c$	$\neg s \wedge \neg c$
b	$p_{11} \times q$	$p_{11} \times (1-q)$	$p_{12} \times q$	$p_{12} \times (1-q)$
$\neg b$	$p_{21} \times q$	$p_{22} \times (1-q)$	$p_{22} \times q$	$p_{22} \times (1-q)$

Table 3: Contingency Table for Case M_2

	$s \wedge c$	$s \wedge \neg c$	$\neg s \wedge c$	$\neg s \wedge \neg c$
b	$p \times q_{11}$	$p \times q_{12}$	$(1-p) \times q_{11}$	$(1-p) \times q_{12}$
$\neg b$	$p \times q_{21}$	$p \times q_{22}$	$(1-p) \times q_{21}$	$(1-p) \times q_{22}$

Table 4: Contingency Table for Case M_3

	$s \wedge c$	$s \wedge \neg c$	$\neg s \wedge c$	$\neg s \wedge \neg c$
b	p_{11}	p_{12}	p_{13}	p_{14}
$\neg b$	p_{21}	p_{22}	p_{23}	p_{24}

- n_{14} : The frequency of when $\neg s$ is followed by $\neg c$, and b is a word segmentation.
- n_{21} : The frequency of when s is followed by c, and b is not a word segmentation.
- n_{22} : The frequency of when s is followed by $\neg c$, and b is not a word segmentation.
- n_{23} : The frequency of when $\neg s$ is followed by c, and b is not a word segmentation.
- n_{24} : The frequency of when $\neg s$ is followed by $\neg c$, and b is not a word segmentation.
- n: The total number of all observed combinations of s, c, and b. $(n_{11} + n_{12} + n_{13} + n_{14} + n_{21} + n_{22} + n_{23} + n_{24})$

Given these values, we then compare 4 hypotheses, being: s relates to b being a word segmentation (M_1), c relates to b being a word segmentation (M_2), both s and c relates to b being a word segmentation (M_3), or nor s and c relates to b being a word segmentation (M_0). We illustrate the formula for each model M_1, M_2, M_3 and M_0 as follows:

(M_1): s relates to b being a word segmentation

The possibility of s deciding whether b is a word segmentation can be represented as variables p_{11}, p_{12}, p_{21}, p_{22}. The values of variables $p_{11}, p_{12}, p_{21}, p_{22}$ are decided by variable q. In this case, $p_{11} + p_{12} + p_{21} + p_{22} = 1$. The 2x4 contingency table is as shown in Table 2.

$$\begin{aligned} MLL(M_1) = \\ &(n_{11} + n_{12}) \log(n_{11} + n_{12}) + (n_{13} + n_{14}) \log(n_{13} + n_{14}) \\ &+ (n_{21} + n_{22}) \log(n_{21} + n_{22}) + (n_{23} + n_{24}) \log(n_{23} + n_{24}) \\ &+ (n_{11} + n_{13} + n_{21} + n_{23}) \log(n_{11} + n_{13} + n_{21} + n_{23}) \\ &+ (n_{12} + n_{14} + n_{22} + n_{24}) \log(n_{12} + n_{14} + n_{22} + n_{24}) \\ &- 2n \log n \end{aligned} \tag{1}$$

$$AIC(M_1) = -2 \times MLL + 2 \times 4 \tag{2}$$

(M_2): c relates to b being a word segmentation

The possibility of c deciding whether b is a word segmentation can be represented as variables q_{11}, q_{12}, q_{21}, q_{22}. Variables $q_{11}, q_{12}, q_{21}, q_{22}$ are decided by variable p. In this case, $q_{11} + q_{12} + q_{21} + q_{22} = 1$. The 2x4 contingency table is as shown in Table 3.

$$\begin{aligned} MLL(M_2) = \\ &(n_{11} + n_{12} + n_{21} + n_{22}) \log(n_{11} + n_{12} + n_{21} + n_{22}) \\ &+ (n_{13} + n_{14} + n_{23} + n_{24}) \log(n_{13} + n_{14} + n_{23} + n_{24}) \\ &+ (n_{11} + n_{13}) \log(n_{11} + n_{13}) + (n_{12} + n_{14}) \log(n_{12} + n_{14}) \\ &+ (n_{21} + n_{23}) \log(n_{21} + n_{23}) + (n_{22} + n_{24}) \log(n_{22} + n_{24}) \\ &- 2n \log n \end{aligned} \tag{3}$$

$$AIC(M_2) = -2 \times MLL + 2 \times 4 \tag{4}$$

(M_3): Both s and c relates to b being a word segmentation

The possibility of both s and c deciding whether b is a word segmentation can be represented as variables p_{11}, p_{12}, p_{21}, p_{22} and p_{21}, p_{22}, p_{23}, p_{22}. In this case, $p_{11} + p_{12} + p_{13} + p_{14} + p_{21} + p_{22} + p_{23} + p_{24} = 1$. The 2x4 contingency table is as shown in Table 4.

$$\begin{aligned} MLL(M_3) = \\ &n_{11} \log n_{11} + n_{12} \log n_{12} + n_{13} \log n_{13} + n_{14} \log n_{14} \\ &+ n_{21} \log n_{21} + n_{22} \log n_{22} + n_{23} \log n_{23} + n_{24} \log n_{24} \\ &- n \log n \end{aligned} \tag{5}$$

$$AIC(M_3) = -2 \times MLL + 2 \times 7 \tag{6}$$

(M_0): Nor s or c, separate or combined, are related to b

Hypothesis: The presence of s, c, and $s \wedge c$ are determined by individual variables p, q, r. The 2x4 contingency table is as shown in Table 5.

Table 5: Contingency Table for Case M_0

	$s \wedge c$	$s \wedge \neg c$	$\neg s \wedge c$	$\neg s \wedge \neg c$
C	$p \times q \times r$	$p \times q \times (1-r)$	$p \times (1-q) \times r$	$p \times (1-q) \times (1-r)$
$\neg C$	$(1-p) \times q \times r$	$(1-p) \times q \times (1-r)$	$(1-p) \times (1-q) \times r$	$(1-p) \times (1-q) \times (1-r)$

$$
\begin{aligned}
MLL(M_0) = & \\
& (n_{11}+n_{12}+n_{13}+n_{14}) \log(n_{11}+n_{12}+n_{13}+n_{14}) \\
& + (n_{11}+n_{12}+n_{21}+n_{22}) \log(n_{11}+n_{12}+n_{21}+n_{22}) \\
& + (n_{11}+n_{13}+n_{21}+n_{23}) \log(n_{11}+n_{13}+n_{21}+n_{23}) \\
& + (n_{21}+n_{22}+n_{23}+n_{24}) \log(n_{21}+n_{22}+n_{23}+n_{24}) \\
& + (n_{13}+n_{14}+n_{23}+n_{24}) \log(n_{13}+n_{14}+n_{23}+n_{24}) \\
& + (n_{12}+n_{14}+n_{22}+n_{24}) \log(n_{12}+n_{14}+n_{22}+n_{24}) \\
& - 3n \log n \qquad (7)
\end{aligned}
$$

$$AIC(M_0) = -2 \times MLL + 2 \times 3 \qquad (8)$$

After calculating $AIC(M_1)$, $AIC(M_2)$, $AIC(M_3)$ and $AIC(M_0)$, we follow the algorithm shown in Figure 2 to select the approriate AIC model and calculate score E. The algorithm searches for a model with the hypothesis that at least one of the three models (M_1, M_2, M_3) has a more higher likeliness than the independent model (M_0). If not, (M_0) is selected and $E = 0$.

If one of the three models is selected, the selected model (M_f) is tested to see if the feature within the model is more related to b being a word segmentation than b not being a word segmenation. In case the selected model fulfull the condition mentioned above, score E is calculated by subtracting $AIC(M_f)$ from $AIC(M_0)$. Otherwise, the algorithm searches for the next most likely model to be applied, until the independent model (M_0) has the highest likeliness.

```
Input:
n11, n12, n13, n14, n21, n22, n23, n24
Output:
m : Selected model
Procedure:
found = false
M = {M0, M1, M2, M3}
while (¬found) do
  m = argmin_{mi∈M} {AIC(M1),...}
  if m = M1 then
    if (n11 + n12) / (n11 + n12 + n21 + n22) >
       (n13 + n14) / (n13 + n14 + n23 + n24) then
      E ← AIC(M0) - AIC(M1)
      found = true
  elseif m = M2 then
    if (n11 + n13) / (n11 + n13 + n21 + n23) >
       (n12 + n14) / (n12 + n14 + n22 + n24) then
      E ← AIC(M0)-AIC(M2)
      found = true
  else if m = M3 then
    if n11 / (n11 + n21) >
       (n12 + n13 + n14) / (n12 + n13 + n14 + n22 + n23 + n24)
then
      E ← AIC(M0)-AIC(M3)
      found = true
  else
      break
  M = M - {m}
```

Figure 2: Selection of AIC model used to calculate score E

3.2 Transformation to SVM feature

We illustrate on how to use the obtained features to train a classifier. In our current implementation, we use a Support Vector Machine[7] (SVM) to determine whether b is a word segmentation. If S is made up of k characters, we must prepare $k-1$ SVM instances, which is impractical when S is a long string. Therefore, we use a "window" scheme which decompose S into smaller strings made up of L characters.

As an example, we show in Figure 3 on how to transform S = "FOOBAR" into smaller windows of the size of L. Note that since we are creating training data, all values of b in the figure is trivial. In order to create features for SVM for b_3 in S, we extract s and c combinations which are related to b_3. In this example, they are the following combinations:

- $s_{11} = c_1 + c_2 + c_3$, $c_{11} = c_4$ ("FOO" + "B")
- $s_{21} = c_2 + c_3 + c_4$, $c_{21} = c_5$ ("OOB" + "A")
- $s_{31} = c_3 + c_4 + c_5$, $c_{31} = c_6$ ("OBA" + "R")

From s_{11}, s_{21}, and s_{31}, we can also create the following combinations:

- $s_{12} = c_2 + c_3$, $c_{12} = c_4$ ("OO" + "B")
- $s_{13} = c_3$, $c_{13} = c_4$ ("O" + "B")
- $s_{22} = c_3 + c_4$, $c_{22} = c_5$ ("OB" + "A")
- $s_{23} = c_4$, $c_{23} = c_5$ ("B" + "A")
- $s_{32} = c_4 + c_5$, $c_{32} = c_6$ ("BA" + "R")
- $s_{33} = c_5$, $c_{33} = c_6$ ("A" + "R")

For each of the 9 string combinations, we assign their associating values: n_{11}, n_{12}, n_{13}, n_{14}, n_{21}, n_{22}, n_{23}, n_{24}, $AIC(M_0)$, $AIC(M_f)$ and E. Finally, we unify the 9 sets of features into a single sample and add the value of b as the

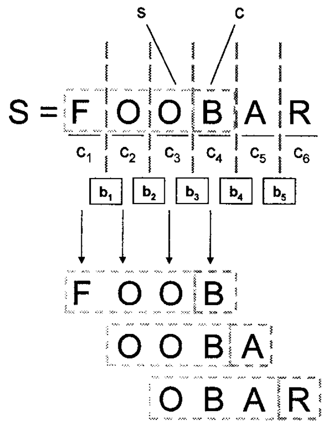

Figure 3: Formulation of window based on L

Table 6: Experiment Results

	2x4 Contingency Table	2x2 Contingency Table
Precision	**0.9525**	0.8536
Recall	**0.9545**	0.8519
f-measure	**0.9534**	0.8527

label for the sample, resulting with a sample with the label b and 99 dimensions, which can be used as training data for an SVM. For the estimation phase, we conduct the same procedure above to transform a given string S into SVM features.

4. Evaluation

For evaluation, we used the Kyoto University Corpus[10], because it is often used as a benchmark in tasks for natural language processing with Japanese. As a classifier, we choosed to use LIBLINEAR[8], an SVM implementation based on linear regression and is aimed to be used for large scale application[9]. For parmeters, we used $c = 1$ for cost, and $L = 4$ for the window size.

The data set of [10] consists of approximately 40,000 documents. We selected 1,000 documents randomly from the data set to use as testing data, and the remaining data set as training data. We conducted this process 3 times, and used the average value of precision and recall to measure the accuracy.

The result is shown in Table 6. The accuracy is measured by using precision, recall and accuracy. We found the proposed method was accurate by 0.95 in both precision and recall. In turn, the conventional method was accurate by 0.85 in both precision and recall. From this, it can be said that creating features based on the co-occurance of a given pair of strings and also the existence of a word segmentation is more accurate by 0.1 in F-measure than judging from only the co-occurrence of the pair of strings.

5. Conclusion

In this paper, we proposed a word segmentation scheme which uses feature selection method with an n-gram model. Unlike conventional methods which use a 2x2 contingency table to calculate the relationship of a pair of strings, our proposal uses a 2x4 contingency table to evaluate the relationship of a pair of strings and also whether there is a word segmentation between them. Using a newspaper corpus, we achieved 0.95 in both precision and recall. We found that the 2x4 contingency table created more accurate features than using an 2x2 contingency table, which resulted with a improvement in F-measure by 0.1.

Acknowledgment

The authors would like to thank Dr. Yasuhiko Itou, Dr. Shigeyuki Akiba, Dr. Shuichi Matsumoto, and Dr. Fumiaki Sugaya from KDDI R&D Laboratories for their comments. Part of this work is supported by the National Institute of Information and Communications Technology (NiCT).

References

[1] JUMAN. http://www-lab25.kuee.kyoto-u.ac.jp/nl-resource/juman.html

[2] ChaSen legacy – an old morphological analyzer http://chasen-legacy.sourceforge.jp/

[3] MeCab: Yet Another Part-of-Speech and Morphological Analyzer: http://mecab.sourceforge.net/MeCab

[4] T. Yanagihara, K. Ikeda, K. Matsumoto, Y. Takishima. "Word Segmentation Estimation using Information Criteria." In Proceedings of the 190th IPSJ Workshop on Natural Language Processing (IPSJ-NL-190), pp.43-48, 2009. *(In Japanese)*

[5] K. Matsumoto and K. Hashimoto. "Schema Design for Casual Law Mining from Incomplete Database." In Discovery Science, Second International Conference, Vol. Lecture Notes in Computer Science 1721 Springer, pp.92-102, 1999.

[6] H. Akaike. "A New Look at the Statistical Model Identification" IEEE Transactions on Automatic Control 19 (6): pp. 716-723, 1974

[7] V. N. Vapnik. "The Nature of Statistical Learning Theory." Springer-Verlag. 1995.

[8] LIBLINEAR – A Library for Large Linear Classification. http://www.csie.ntu.edu.tw/ cjlin/liblinear/

[9] C. J. Hsieh, K. W. Chang, C. J. Lin, S. S. Keerthi, and S. Sundararajan. "A Dual Coordinate Descent Method for Large-scale Linear SVM". Proceedings of the 25 th International Conference on Machine Learning (ICML2008), 2008.

[10] Kyoto University Text Corpus. http://www-lab25.kuee.kyoto-u.ac.jp/nl-resource/corpus.html EDR Corpus. http://www2.nict.go.jp/r/r312/EDR/J_index.html