JAPANESE WORD SEGMENTATION BY HIDDEN MARKOV MODEL

Constantine P. Papageorgiou

BBN Systems and Technologies
70 Fawcett St.
Cambridge, MA 02138

ABSTRACT

The processing of Japanese text is complicated by the fact that there are no word delimiters. To segment Japanese text, systems typically use knowledge-based methods and large lexicons. This paper presents a novel approach to Japanese word segmentation which avoids the need for Japanese word lexicons and explicit rule bases. The algorithm utilizes a hidden Markov model, a stochastic process, to determine word boundaries. This method has achieved 91% accuracy in segmenting words in a test corpus.

1. INTRODUCTION

The segmentation of Japanese words is one of the main challenges in the automatic processing of Japanese text. Unlike English text which has spaces that separate consecutive words, there are no such word boundary indicators in sentences of Japanese (kanji and kana) text.

The algorithms used to obtain robust segmentation of Japanese text generally utilize two techniques, lexicon and rule-based approaches. Large lexicons are inevitably used in conjunction with or as a part of the text segmenting algorithms that have been developed. These lexicons are often time consuming to build and are thus not an optimal solution. Knowledge based approaches typically entail a significant amount of human effort in specifying the rules that will be used to determine word segmentation and do not provide sufficient coverage of the language's grammar rules.

This paper introduces a hidden Markov model (HMM) which has been developed for Japanese word segmentation. Hidden Markov models are part of the larger class of probabilistic algorithms. These approaches use large sets of data to abstract away the structure of the domain being learned, as probabilities. We will see that with sufficient data such a stochastic process can achieve 91% accuracy in word segmentation which approaches the state-of-the-art in segmentation techniques.

2. JAPANESE WORD SEGMENTATION TECHNIQUES

Current Japanese word segmentation techniques consistently rely on large lexicons of words in their decision making procedure. A typical word processor will have both a knowledge base which encodes rules of Japanese grammar, as well as a lexicon of over 50,000 words [Mori, et. al. 1990]. Hypothesized segments of incoming text are analyzed to determine if they have any semantics and therefore are more likely to be correct segments; the grammar rules are then invoked to make final segmentation decisions. This technology achieves 95 percent accuracy in segmentation.

An alternative approach to Japanese word processing technology is the development of an architecture for Japanese segmentation and part of speech labeling shown in Figure 1 [Matsukawa, et. al. 1993].

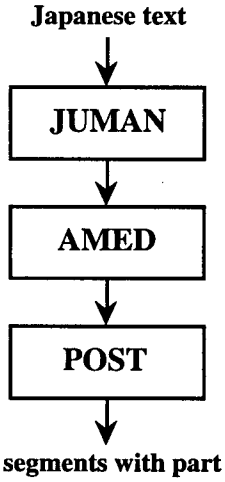

Figure 1: BBN's JUMAN/AMED/POST word segmentation and part of speech labeling architecture.

The architecture of the system is as follows:

1. JUMAN, a rule-based morphological processor which uses a 40,000 word lexicon and a connectivity matrix to determine word segmentation and part of speech labeling,
2. AMED, a rule-based segmentation and part of speech correction system trained on parallel hypothesized and correct annotations of identical text,
3. POST, a hidden Markov model which disambiguates segmentation and part of speech decisions.

This unified architecture achieved an error rate of 8.3% in word segmentation; this level of error can be attributed to JUMAN's relatively small lexicon and lack of sufficient training data for the AMED and POST modules.

Dragon Systems' LINGSTAT machine translation system [Yamron, et. al., 1993] uses a maximum likelihood segmentation algorithm which, in essence, calculates all possible segmentations of a sentence using a large lexicon and chooses the one with the best score, or likelihood. The implementation uses a dynamic programming algorithm to make this search efficient.

MAJESTY is a recently developed morphological preprocessor for Japanese text [Kitani, et. al., 1993]. On a test corpus, it achieved better than 98% accuracy in word segmentation and part of speech determination; this represents the state-of-the-art in such technology.

Teller, et. al. [1994] present a probabilistic algorithm which uses character type information and bi-gram frequencies on characters in conjunction with a small knowledge base to segment non-kanji strings. While it is related to our hidden Markov model approach in that it is character-based and does not rely on any lexicons, it differs in that it relies on a certain amount of a priori knowledge about the morphology of the Japanese language. This algorithm achieved 94.4% accuracy in segmenting words in a test corpus.

3. HIDDEN MARKOV MODEL

Hidden Markov models are widely used stochastic processes which have two components. The first is an observable stochastic process that produces sequences of symbols from a given alphabet. This process depends on a separate hidden stochastic process that yields a sequence of states. Hidden Markov models can be viewed as finite state machines which generate sequences of symbols by jumping from one state to another and "emitting" an observation at each state.

The general recognition problem, as stated in the literature, is: given a hidden Markov model, M, with n symbols and m states, and a sequence of observations, $O = o_1 o_2 ... o_t$, determine the most likely sequence of states, $S = s_1 s_2 ... s_t$ which could yield the observed sequence of symbols.

3.1. Model Development

The hidden Markov model for Japanese word segmentation was designed with several goals in mind:

- Avoiding an approach which relies on having a large lexicon of Japanese words.
- Allow the model to be easily extensible (with new training, of course) to accommodate more data or a different language.
- While not of paramount importance, an algorithm which segments rapidly would be preferred; word segmentation is a pre-processing step in most Japanese text systems and should be as unobtrusive and transparent as possible.

One possible algorithm for segmentation is to use a hidden Markov model to find the most likely sequence of words based on a brute force computation of every possible sequence of words; the POST component of the word segmentation architecture described in Section 2 uses a similar model. This, though, violates the above constraint of no reliance on a Japanese word lexicon. Given that we would like to avoid the overhead associated with constructing and using a word-based lexicon, we are therefore forced to approach the problem in a manner which focuses on discrete characters and their interrelationships.

The segmentation model we developed avoids the need for both a lexicon of Japanese words and explicit rules. It takes advantage of the effectiveness of subsequences of two text characters in determining the presence or absence of a word boundary. In essence, we will show that the morphology of the Japanese language is such that 2-character sequences have some underlying meaning or significance with respect to word boundaries.

To solidify this idea, let us focus on two unspecified text characters, k_1 and k_2. Suppose that out of 100 places in the training data where k_1 is followed by k_2, the vast majority of these occur at word boundaries. From a probabilistic viewpoint, we are justified in coming to the conclusion that "k_1k_2 denotes a word boundary". To complicate things, assume that out of 100 places where k_1 is followed by k_2, 50 of these are at word boundaries and 50 are inside words. It would seem that no conclusions could be drawn from this situation. On the other hand, if we notice that the 50 instances of k_1k_2 at word boundaries all had word boundaries between k_1 and the character preceding k_1, but none of the instances of k_1k_2 within a word had word boundaries before the k_1, then we can hypothesize the following relationship, where "|" denotes a word boundary and k_x is the character preceding k_1:

$$\text{if } k_x \mid k_1, \text{ then } k_1 \mid k_2 \text{ otherwise } k_1k_2$$

This is exactly the sort of hidden structure that HMMs are geared towards uncovering.

Proceeding in this manner, a model for Japanese word segmentation was developed which capitalizes on this idea of the significance of 2-character sequences in word boundary determination; the state transition diagram is shown in Figure 2. In the model there are just two possible states, either a word boundary (B) or a word continuation (C). The observation symbols are all possible 2-character sequences The kanji alphabet consists of approximately 50,000 characters; of these, 6,877 form a standard character set which suits most text processing purposes [Miyazawa, 1990; Mori, et. al. 1990]. Factoring in the size of the hiragana and katakana alphabets, the number of possible 2-character sequences generated exclusively by this subset approaches $5*10^7$, a clearly unmanageable amount of data. An implicit assumption

of our model is that there is a small subset of all possible 2-character sequences which in fact accounts for a large percentage of the 2-character sequences normally used in written text. It is such a subset which the model hopes to uncover and use in further classification.

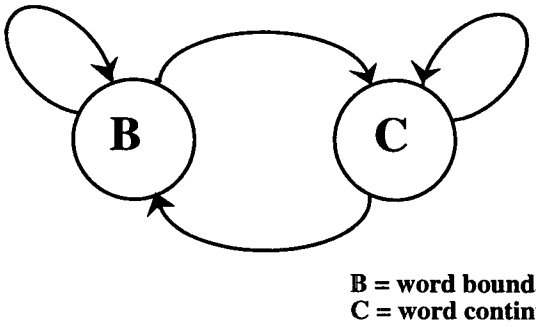

Figure 2: The state transition diagram for the Japanese word segmentation HMM.

The algorithm proceeds by sliding a 2-character window over an input sentence and calculating how likely it is that each of these 2-character sequences is a word boundary or within a word, given the previous 2-character sequence's status as either a word boundary or continuation. In this manner, the model is a bi-gram model [Meteer, et. al., 1991] over 2-character sequences since it relies only on the previous state. It is important to note that consecutive 2-character windows overlap by one character. Figure 3 portrays the progression of the window across part of a line of text emitting the 2-character observation symbols.

3.2. Training

The model is trained using supervised training over a previously annotated corpus. Specifically, training is accomplished by taking the corpus of segmented text and simply counting the number of times each 2-character sequence

- has a word boundary between its constituent characters
- has no word boundary between its constituent characters

and the number of times

- word boundaries follow word continuations
- word boundaries follow word boundaries
- word continuations follow word boundaries
- word continuations follow word continuations

Seeing unknown 2-character sequences in the test data (those sequences that were absent from the training data) leads to observation probabilities of 0. To rectify this, upon coming across an unknown 2-character sequence in the test data, the algorithm assigns the observation an a priori probability by postulating that the sequence was actually seen once in the training data. This probability is a sufficiently low value, balancing the fact that the sequence was never seen when all possible symbols were being gathered, with the hypothesis that it might be a valid observation. This procedure is an admission that even extensive training might not attain complete coverage of the domain.

3.4. Implementation Issues

Algorithm -- There are generally two basic algorithms for hidden Markov model recognition: the forward-backward algorithm and the Viterbi algorithm [Viterbi, 1967]. The forward-backward algorithm (Baum-Welch) computes the likelihood that the sequence of observation symbols was produced by any possible state sequence. The Viterbi model, on the other hand, computes the likelihoods based on the best possible state sequence and is more efficient to compute and train. The word segmentation HMM implementation uses the Viterbi approach. This difference is transparent and matters only at the implementation level.

Kanji and Kana -- Due to their vast numbers, two bytes are needed to represent Japanese text characters rather than the conventional one byte for English characters. The implementation can easily support either one or two byte characters with few modifications.

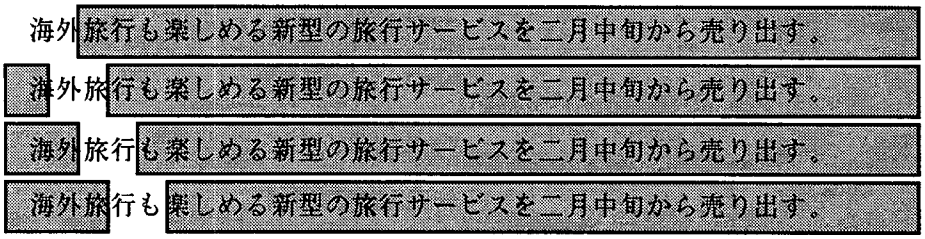

Figure 3: Diagram shows the 2-character window sliding over the sentence and uncovering the first four observations.

Sentence by Sentence Input -- The only assumption on the input to the hidden Markov model is that the text be pre-divided into sentences. Periods were the sole indicators of sentence endings that were used. This assumption is made to provide for the incremental processing of a body of text.

4. EXPERIMENTS AND ANALYSIS

To train and test the hidden Markov model, a corpus of 5,529 Japanese articles that was annotated by the MAJESTY system was used since a manually annotated corpus of sufficient size was not available. From these articles, 59,587 sentences (1,882,231 words) were used as training material and 634 different sentences (21,430 words) were set aside as test data. When the trained model was run over the test sentences, it segmented 91.15% of the words correctly while achieving 96.48% accuracy on word boundaries. The correct segmentation of a single word implies that:

- both its beginning and ending word boundaries are determined correctly, and
- no extra word boundaries are generated within the word.

The results over distinct words are given in Table 1 and the results for word boundaries are in Table 2.

	Number	% of total
total	21,430	-
hypothesized	21,298	-
correct	19,533	91.15
incorrect	1,897	8.85

Table 1: Test results over words.

	Number	% of total
total	20,796	-
hypothesized	20,664	-
correct	20,065	96.48
overgenerated	599	2.88
undergenerated	731	3.52

Table 2: Test results over word boundaries.

These performance figures compare favorably with the previously reported results of the BBN Japanese word segmentation and part of speech algorithm. This system, described in Section 2 and currently in use in the BBN PLUM data extraction system, achieved 91.7% accuracy in word segmentation in a test. In addition, the word segmentation HMM was designed and implemented in under one person-week, whereas the aforementioned architecture and all its components took significantly longer.

The performance figures listed above are telling; with a simple but cleverly constructed model, the system managed to correctly segment words at a respectable rate. This performance was achieved entirely without accessing any of the word lexicons that are traditionally employed in solving this problem. Furthermore, no rule bases are referred to; the algorithm simply relies on the structure of the training data to implicitly obtain a model of Japanese word segmentation.

While the HMM both misses and imagines word boundaries, it is encouraging that the total numbers of hypothesized words and word boundaries are close to the true numbers. This assures us that the model is generating an appropriate number of boundaries, even though it is not completely accurate on all of them.

The fact that the model performs to such a high degree has interesting implications regarding the morphology of the Japanese language. The model relies on the idea that consecutive characters are significant with regards to whether or not they will be separated by a word boundary. This suggests that there is a set of pairs of characters which rarely occur next to one another within the same word; these are the 2-character boundary sequences used in the HMM and include at least the katakana character set as an edge. Furthermore, there must be another set of character pairs which are frequently found in succession in the same word, corresponding to the model's 2-character continuation sequences.

4.1. Training Set Size

As with any stochastic model, this HMM relies on an accurate set of probabilities which reflect the true nature of the domain. The limiting factor here, barring any gross problems with the model, is the amount of data on which the model is trained. Clearly, when the training procedure sees the first few examples, the HMM is a very poor representation of Japanese word boundaries. As such, a large amount of information is collected in a relatively short period of time in the initial stages of learning. The model will eventually become more complete as it sees a larger and larger portion of the possible 2-character sequences.

Determining where the size of the training set no longer seems to be having a great impact on the performance of the algorithm is of interest as we can find out if the model is under-trained or over-trained. To get a sense for this, the model was trained on successively larger test sets, starting with a very small training set of 123 words up to the 1,882,231 word set, and then run over the 21,430 word test set and evaluated. Figure 4 summarizes the results of these experiment.

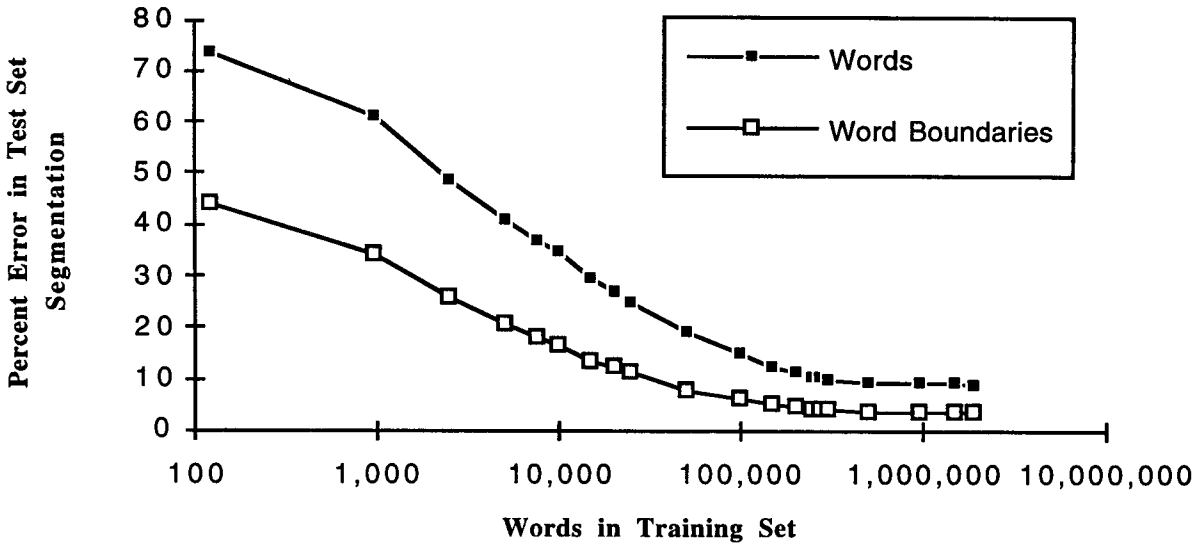

Figure 4: Effect of training set size on performance.

Using a logarithmic scale for the axis representing training set size gives a feeling for the additional performance accrued from more training, while factoring in the impact of the exponentially increasing advances in computing technology. Based on the graph, we can see that while the word segmentation error rate is diminishing more slowly as the training set size increases to 1,882,231 words (the final point plotted), the curve still exhibits a downward trend. This implies that additional training could improve the accuracy of this model.

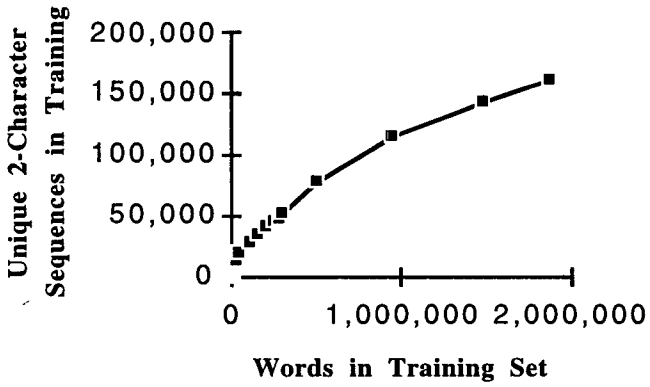

Figure 5: Number of unique 2-character sequences.

As expected, the largest increase in performance occurs over the initial 30,000 words where the word segmentation error rate goes from 75% to 25%. At approximately 150,000 words, the rate of change in the error rate decreases significantly, but still shows a distinct downward trend. Furthermore, the difference between the word segmentation error rate and word boundary determination error rate is continuously shrinking; it is expected that with additional training data the gap between the curves will diminish.

To portray the amount of new information that is received over time, Figure 5 shows the number of unique 2-character sequences in each of the successively increasing training sets. It is interesting to note that the model is continuously seeing new 2-character sequences at a steady, though slightly decreasing, rate. By the time the training set numbers 50,000 words, the most common 2-character sequences have been seen and further training data, while improving test performance, provides diminishing returns due to the relative rarity of these new sequences.

5. CONCLUSION

We have implemented and described a hidden Markov model for Japanese word segmentation. The bi-gram model is characterized by an unconventional set of observation symbols, namely, the set of 2-character sequences. The model is also extremely simple in that it consists of only two states which encode the existence or absence of a word boundary between any two characters. This probabilistic model was trained over a large corpus of annotated data and then tested over a different set of data to measure performance; it achieves word segmentation accuracy of 91.15% and determines 96.48% of all the word boundaries correctly. When contrasted with the state-of-the-art, the HMM emerges as a worthy contender to related algorithms based on several observations:

1. First and foremost, this HMM approach completely circumvents the need for Japanese word lexicons which other approaches heavily rely upon; the storage issues and overhead for word look-up are thus avoided.

2. The rules that a knowledge-based system would use are, in effect, implicit in the probabilities determined during supervised training and exactly reflect the morphology of Japanese word boundaries.

3. The HMM segments text at a blistering pace, approximately 10,000 words/second not including initialization time.

4. The model is designed to be easily extensible with additional data or to a different language; no lexicons are needed, simply a sufficiently large body of text on which the algorithm can be trained.

Most disappointing about the performance of the model is the large discrepancy between the word accuracy and the word boundary accuracy. This is surely a side-effect of the bi-gram model topology; there is no way to relate the beginning and ending boundaries of a single word with this model unless the word begins and ends in consecutive states (a one-character word).

Regardless, it is interesting and impressive that a two state bi-gram model can model Japanese word boundaries so effectively. With additional training data, we anticipate that the algorithm's performance will increase. The next generation of this model should somehow incorporate and model the relationship between boundaries of the same word in an effort to raise the word segmentation accuracy closer to the accuracy level of word boundary determination. Another modification to the algorithm which might improve performance is extending it to be a tri-gram model. The HMM could also be trained and tested on a different language, Chinese for instance, to see how well it performs.

The results of this research are encouraging; the re-training and extensions noted above should be pursued to increase accuracy and to obtain a sense of how generally applicable to comparable domains this hidden Markov model is.

ACKNOWLEDGEMENTS

The work reported here was supported in part by the Advanced Research Projects Agency and was monitored by the Rome Air Development Center under Contract No. F30602-91-C-0051. The views and conclusions contained in this document are those of the author and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Advanced Research Projects Agency or the United States Government.

REFERENCES

1. Church, K. "A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text". *Proceedings of the Second Conference on Applied Natural Language Processing*, ACL, 1988, p. 136-143.

2. Kitani, T. and Mitamura, T. "Japanese preprocessor for syntactic and semantic parsing". *Proceedings of the Conference on Artificial Intelligence Applications*, 1993, p. 86-92.

3. Matsukawa, T., Miller, S., and Weischedel, R. "Example-Based Correction of Word Segmentation and Part of Speech Labeling". *Human Language Technology*, March 1993, p. 227-232.

4. Meteer, M., Schwartz, R., and Weischedel, R. "Empirical Studies in Part of Speech Labeling". *Proceedings of the Fourth DARPA Workshop on Speech and Natural Language*, February 1991, p. 331-336.

5. Miyazawa, A. "Character Code for Japanese Text Processing". *Journal of Information Processing* 13(1), 1990, p. 2-9.

6. Mori, K. and Kawada, T. "From kana to kanji: word processing in Japan". *IEEE Spectrum*, August 1990, p. 46-48.

7. Teller, V. and Batchelder, E. O. "A Probabilistic Algorithm for Segmenting Non-Kanji Japanese Strings". to appear in *Proceedings of 12th National Conference on Artificial Intelligence*, 1994.

8. Viterbi, A. J. "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm". *IEEE Transactions on Information Theory IT* 13(2), April 1967, pp. 260-269.

9. Yamron, J., Baker, J., Bamberg, P., Chevalier, H., Dietzel, T., Elder, J., Kampmann, F., Mandel, M., Manganaro, L., Margolis, T., and Steele, E. "LINGSTAT: An Interactive, Machine-Aided Translation System". *Human Language Technology*, March 1993, p. 191-195.