Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Philadelphia, July 2002, pp. 281-288. Association for Computational Linguistics.

Combining Outputs of Multiple Japanese Named Entity Chunkers by Stacking

Takehito Utsuro
Department of Information and Computer Sciences,
Toyohashi University of Technology
Tenpaku-cho, Toyohashi 441-8580, Japan
utsuro@ics.tut.ac.jp

Manabu Sassano
Fujitsu Laboratories, Ltd.
4-4-1, Kamikodanaka, Nakahara-ku,
Kawasaki 211-8588, Japan
sassano@jp.fujitsu.com

Kiyotaka Uchimoto
Keihanna Human Info-Communications Research Center,
Communications Research Laboratory
Hikaridai Seika-cho, Kyoto 619-0289, Japan
uchimoto@crl.go.jp

Abstract

In this paper, we propose a method for learning a classifier which combines outputs of more than one Japanese named entity extractors. The proposed combination method belongs to the family of *stacked generalizers*, which is in principle a technique of combining outputs of several classifiers at the first stage by learning a second stage classifier to combine those outputs at the first stage. Individual models to be combined are based on maximum entropy models, one of which always considers surrounding contexts of a fixed length, while the other considers those of variable lengths according to the number of constituent morphemes of named entities. As an algorithm for learning the second stage classifier, we employ a decision list learning method. Experimental evaluation shows that the proposed method achieves improvement over the best known results with Japanese named entity extractors based on maximum entropy models.

1 Introduction

In the recent corpus-based NLP research, system combination techniques have been successfully applied to several tasks such as parts-of-speech tagging (van Halteren et al., 1998), base noun phrase chunking (Tjong Kim Sang, 2000), and parsing (Henderson and Brill, 1999; Henderson and Brill, 2000). The aim of system combination is to combine portions of the individual systems' outputs which are partial but can be regarded as highly accurate. The process of system combination can be decomposed into the following two sub-processes:

1. Collect systems which behave as differently as possible: it would help a lot if at least the collected systems tend to make errors of different types, because simple voting technique can identify correct outputs.

   Previously studied techniques for collecting such systems include: i) using several existing real systems (van Halteren et al., 1998; Brill and Wu, 1998; Henderson and Brill, 1999; Tjong Kim Sang, 2000), ii) bagging/boosting techniques (Henderson and Brill, 1999; Henderson and Brill, 2000), and iii) switching the data expression and obtaining several models (Tjong Kim Sang, 2000).
2. Combine the outputs of the several systems: previously studied techniques include: i) voting techniques (van Halteren et al., 1998; Tjong Kim Sang, 2000; Henderson and Brill, 1999; Henderson and Brill, 2000), ii) switching among several systems according to confidence values they provide (Henderson and Brill, 1999), iii) stacking techniques (Wolpert, 1992) which train a second stage classifier for

combining outputs of classifiers at the first stage (van Halteren et al., 1998; Brill and Wu, 1998; Tjong Kim Sang, 2000).

In this paper, we propose a method for combining outputs of (Japanese) named entity chunkers, which belongs to the family of stacking techniques. In the sub-process 1, we focus on models which differ in the lengths of preceding/subsequent contexts to be incorporated in the models. As the base model for supervised learning of Japanese named entity chunking, we employ a model based on the maximum entropy model (Uchimoto et al., 2000), which performed the best in IREX (Information Retrieval and Extraction Exercise) Workshop (IREX Committee, 1999) among those based on machine learning techniques. Uchimoto et al. (2000) reported that the optimal number of preceding/subsequent contexts to be incorporated in the model is two morphemes to both left and right from the current position. In this paper, we train several maximum entropy models which differ in the lengths of preceding/subsequent contexts, and then combine their outputs.

As the sub-process 2, we propose to apply a stacking technique which learns a classifier for combining outputs of several named entity chunkers. This second stage classifier learns rules for accepting/rejecting outputs of several individual named entity chunkers. The proposed method can be applied to the cases where the number of constituent systems is quite small (e.g., two). Actually, in the experimental evaluation, we show that the results of combining the best performing model of Uchimoto et al. (2000) with the one which performs poorly but extracts named entities quite different from those of the best performing model can help improve the performance of the best model.

2 Named Entity Chunking based on Maximum Entropy Models

2.1 Task of the IREX Workshop

The task of named entity recognition of the IREX workshop is to recognize eight named entity types in Table 1 (IREX Committee, 1999). The organizer of the IREX workshop provided 1,174 newspaper articles which include 18,677 named entities as the training data. In the formal run (general domain) of the workshop, the participating systems were requested to recognize 1,510 named entities included in the held-out 71 newspaper articles.

Table 1: Statistics of NE Types of IREX

NE Type	frequency (%)	
	Training	Test
ORGANIZATION	3676 (19.7)	361 (23.9)
PERSON	3840 (20.6)	338 (22.4)
LOCATION	5463 (29.2)	413 (27.4)
ARTIFACT	747 (4.0)	48 (3.2)
DATE	3567 (19.1)	260 (17.2)
TIME	502 (2.7)	54 (3.5)
MONEY	390 (2.1)	15 (1.0)
PERCENT	492 (2.6)	21 (1.4)
Total	18677	1510

2.2 Named Entity Chunking

We first provide our definition of the task of Japanese named entity chunking (Sekine et al., 1998; Borthwick et al., 1998; Uchimoto et al., 2000). Suppose that a sequence of morphemes is given as below:

$$\underset{\text{(Left Context)}}{\cdots M^L_{-k} \cdots M^L_{-1}} \quad \underset{\text{(Named Entity)}}{M^{NE}_1 \cdots M^{NE}_i \cdots M^{NE}_m} \quad \underset{\text{(Right Context)}}{M^R_1 \cdots M^R_l \cdots}$$

$$\uparrow$$

(Current Position)

Given that the current position is at the morpheme M^{NE}_i, the task of named entity chunking is to assign a chunking state (to be described in section 2.3.1) to the morpheme M^{NE}_i at the current position, considering the patterns of surrounding morphemes. Note that in the supervised learning phase, we can use the chunking information on which morphemes constitute a named entity, and which morphemes are in the left/right contexts of the named entity.

2.3 The Maximum Entropy Model

In the maximum entropy model (Della Pietra et al., 1997), the conditional probability of the output y given the context x can be estimated as the following $p_\lambda(y \mid x)$ of the form of the exponential family, where binary-valued indicator functions called *feature functions* $f_i(x, y)$ are introduced for expressing a set of "features", or "attributes" of the context x and the output y. A *parameter* λ_i is introduced for each feature f_i, and is estimated from a training data.

$$p_\lambda(y \mid x) = \frac{\exp\Big(\sum_i \lambda_i f_i(x, y)\Big)}{\sum_y \exp\Big(\sum_i \lambda_i f_i(x, y)\Big)}$$

Uchimoto et al. (2000) defines the context x as the patterns of surrounding morphemes as well as that at the current position, and the output y as the named entity chunking state to be assigned to the morpheme at the current position.

2.3.1 Named Entity Chunking States

Uchimoto et al. (2000) classifies classes of named entity chunking states into the following 40 tags:

- Each of eight named entity types plus an "OPTIONAL" type are divided into four chunking states, namely, the beginning/middle/end of an named entity, or an named entity consisting of a single morpheme. This amounts to $9 \times 4 = 36$ classes.
- Three more classes are distinguished for morphemes immediately preceding/following a named entity, as well as the one between two named entities.
- Other morphemes are assigned the class "OTHER".

2.3.2 Features

Following Uchimoto et al. (2000), feature functions for morphemes at the current position as well as the surrounding contexts are defined. More specifically, the following three types of feature functions are used: [1]

1. 2052 lexical items that are observed five times or more within two morphemes from named entities in the training corpus.
2. parts-of-speech tags of morphemes[2].
3. character types of morphemes (i.e., Japanese (hiragana or katakana), Chinese (kanji), numbers, English alphabets, symbols, and their combinations).

As for the number of preceding/subsequent morphemes as contextual clues, we consider the following models:

[1]Minor modifications from those of Uchimoto et al. (2000) are: i) we used character types of morphemes because they are known to be useful in the Japanese named entity chunking, and ii) the sets of parts-of-speech tags are different.

[2]As a Japanese morphological analyzer, we used BREAKFAST (Sassano et al., 1997) with the set of about 300 part-of-speech tags. BREAKFAST achieves 99.6% part-of-speech accuracy against newspaper articles.

5-gram model

This model considers the preceding two morphemes M_{-2}, M_{-1} as well as the subsequent two morphemes M_1, M_2 as the contextual clue. Both in (Uchimoto et al., 2000) and in this paper, this is the model which performs the best among all the individual models without system combination.

(Left Context)	(Current Position)	(Right Context)
$\cdots\ M_{-2}\ M_{-1}$	M_0	$M_1\ M_2\ \cdots$

7-gram model

This model considers the preceding three morphemes M_{-3}, M_{-2}, M_{-1} as well as the subsequent three morphemes M_1, M_2, M_3 as the contextual clue.

(Left Context)	(Current Position)	(Right Context)
$\cdots\ M_{-3}\ M_{-2}\ M_{-1}$	M_0	$M_1\ M_2\ M_3\ \cdots$

9-gram model

This model considers the preceding four morphemes $M_{-4}, M_{-3}, M_{-2}, M_{-1}$ as well as the subsequent four morphemes M_1, M_2, M_3, M_4 as the contextual clue.

(Left Context)	(Current Position)	(Right Context)
$\cdots\ M_{-4} \cdots M_{-1}$	M_0	$M_1 \cdots M_4\ \cdots$

For both 7-gram and 9-gram models, we consider the following three modifications to those models:

- with all features
- with lexical items and parts-of-speech tags (without the character types) of $M_{\{(-4),-3,3,(4)\}}$
- with only the lexical items of $M_{\{(-4),-3,3,(4)\}}$

In our experiments, the number of features is 13,200 for 5-gram model and 15,071 for 9-gram model. The number of feature functions is 31,344 for 5-gram model and 35,311 for 9-gram model.

Training a variable length (5∼9-gram) model, testing with 9-gram model

The major disadvantage of the 5/7/9-gram models is that in the training phase it does not take into account whether or not the preceding/subsequent morphemes constitute one named entity together with the morpheme at the current position. Considering this disadvantage, we examine another model, namely, *variable length model*, which incorporates variable length contextual information. In the training phase, this model considers which of the preceding/subsequent morphemes constitute one named

entity together with the morpheme at the current position (Sassano and Utsuro, 2000). It also considers several morphemes in the left/right contexts of the named entity. Here we restrict this model to explicitly considering the cases of named entities of the length up to three morphemes and only implicitly considering those longer than three morphemes. We also restrict it to considering two morphemes in both left and right contexts of the named entity.

(Left Context)	(Named Entity)	(Right Context)
$\cdots\ M_{-2}^{L} M_{-1}^{L}$	$M_1^{NE} \cdots M_i^{NE} \cdots M_{m(\leq 3)}^{NE}$	$M_1^{R} M_2^{R}\ \cdots$
	$\uparrow$	
	(Current Position)	

1. In the cases where the current named entity consists of up to three morphemes, all the constituent morphemes are regarded as within the current named entity. The following is an example of this case, where the current named entity consists of three morphemes, and the current position is at the middle of those constituent morphemes as below:

(Left Context)	(Named Entity)	(Right Context)
$\cdots\ M_{-2}^{L} M_{-1}^{L}$	$M_1^{NE} M_2^{NE} M_3^{NE}$	$M_1^{R} M_2^{R}\ \cdots$
	$\uparrow$	(1)
	(Current Position)	

2. In the cases where the current named entity consists of more than three morphemes, only the three constituent morphemes are regarded as within the current named entity and the rest are treated as if they were outside the named entity. For example, suppose that the current named entity consists of four morphemes:

(Left Context)	(Named Entity)	(Right Context)
$\cdots\ M_{-2}^{L} M_{-1}^{L}$	$M_1^{NE} M_2^{NE} M_3^{NE} M_4^{NE}$	$M_1^{R} M_2^{R}\ \cdots$
	$\uparrow$	
	(Current Position)	

   In this case, the fourth constituent morpheme M_4^{NE} is treated as if it were in the right context of the current named entity as below:

(Left Context)	(Named Entity)	(Right Context)
$\cdots\ M_{-2}^{L} M_{-1}^{L}$	$M_1^{NE} M_2^{NE} M_3^{NE}$	$M_4^{NE} M_1^{R}\ \cdots$
	$\uparrow$	
	(Current Position)	

In the testing phase, we apply this model considering the preceding four morphemes as well as the subsequent four morphemes at every position, as in the case of 9-gram model[3].

We consider the following three modifications to this model, where we suppose that the morpheme at the current position be M_0:

- with all features
- with lexical items and parts-of-speech tags (without the character types) of $M_{\{-4,-3,3,4\}}$
- with only the lexical items of $M_{\{-4,-3,3,4\}}$

3 Learning to Combine Outputs of Named Entity Chunkers

3.1 Data Sets

The following gives the training and test data sets for our framework of learning to combine outputs of named entity chunkers.

1. TrI: training data set for learning individual named entity chunkers.
2. TrC: training data set for learning a classifier for combining outputs of individual named entity chunkers.
3. Ts: test data set for evaluating the classifier for combining outputs of individual named entity chunkers.

3.2 Procedure

The following gives the procedure for learning the classifier to combine outputs of named entity chunkers using TrI and TrC.

1. Train the individual named entity chunkers $NEchk_i\ (i = 1, \ldots, n)$ using TrI.
2. Apply the individual named entity chunkers $NEchk_i\ (i = 1, \ldots, n)$ to TrC, respectively, and obtain the list of chunked named entities $NEList_i(TrC)$ for each named entity chunker $NEchk_i$.

[3]Note that, as opposed to the training phase, the length of preceding/subsequent contexts is fixed in the testing phase of this model. Although this discrepancy between training and testing damages the performance of this single model (section 4.1), it is more important to note that this model tends to have distribution of correct/over-generated named entities different from that of the 5-gram model. In section 4, we experimentally show that this difference is the key to improving the named entity chunking performance by system combination.

Table 2: Examples of Event Expressions for Combining Outputs of Multiple Systems

Segment	Morpheme(POS)	NE Outputs of Individual Systems		Event Expressions
		System 0	System 1	
	⋮			
$SegEv_i$	rainen (*"next year"*, temporal_noun) 10gatsu (*"October"*, temporal_noun)	rainen (DATE) 10gatsu (DATE)	rainen -10gatsu (DATE)	$\{systems=\langle 0\rangle, mlength=1, NEtag=\text{DATE}, POS=\langle\text{temporal\_noun}\rangle, class_{NE}=-\}$ $\{systems=\langle 0\rangle, mlength=1, NEtag=\text{DATE}, POS=\langle\text{temporal\_noun}\rangle, class_{NE}=-\}$ $\{systems=\langle 1\rangle, mlength=2, NEtag=\text{DATE}, POS=\langle\text{temporal\_noun, temporal\_noun}\rangle, class_{NE}=+\}$
	⋮			
$SegEv_{i+1}$	seishoku (*"reproductive"*, noun) iryou (*"medical"*, noun) gijutsu (*"technology"*, noun)		seishoku -iryou -gijutsu (ARTIFACT)	$\{systems=\langle 0\rangle, class_{sys}=\text{"no outputs"}\}$ $\{systems=\langle 1\rangle, mlength=3, NEtag=\text{ARTIFACT}, POS=\langle\text{noun,noun,noun}\rangle, class_{NE}=-\}$
	nitsuite (*"about"*, particle) ⋮			

3. Align the lists $NEList_i(TrC)$ $(i=1,\ldots,n)$ of chunked named entities according to the positions of the chunked named entities in the text TrC, and obtain the event expression $TrCev$ of TrC.
4. Train the classifier $NEchk_{cmb}$ for combining outputs of individual named entity chunkers using the event expression $TrCev$.

The following gives the procedure for applying the learned classifier to Ts.

1. Apply the individual named entity chunkers $NEchk_i$ $(i=1,\ldots,n)$ to Ts, respectively, and obtain the list of chunked named entities $NEList_i(Ts)$ for each named entity chunker $NEchk_i$.
2. Align the lists $NEList_i(Ts)$ $(i=1,\ldots,n)$ of chunked named entities according to the positions of the chunked named entities in the text Ts, and obtain the event expression $Tsev$ of Ts.
3. Apply $NEchk_{comb}$ to $Tsev$ and evaluate its performance.

3.3 Data Expressions

3.3.1 Events

The event expression $TrCev$ of TrC is obtained by aligning the lists $NEList_i(TrC)$ $(i=1,\ldots,n)$ of chunked named entities, and is represented as a sequence of segments, where each segment is a set of aligned named entities. Chunked named entities are aligned under the constraint that those which share at least one constituent morpheme have to be aligned into the same segment. Examples of segments, into which named entities chunked by two systems are aligned, are shown in Table 2. In the first segment $SegEv_i$, given the sequence of the two morphemes, the system No.0 decided to extract two named entities, while the system No.1 chunked the two morphemes into one named entity. In those event expressions, $systems$ indicates the list of the indices of the systems which output the named entity, $mlength$ gives the number of the constituent

morphemes, $NEtag$ gives one of the nine named entity types, POS gives the list of parts-of-speech of the constituent morphemes, and $class_{NE}$ indicates whether the named entity is a *correct* one compared against the gold standard ("+"), or the one over-generated by the systems ("−").

In the second segment $SegEv_{i+1}$, only the system No.1 decided to extract a named entity from the sequence of the three morphemes. In this case, the event expression for the system No.0 is the one which indicates that no named entity is extracted by the system No.0.

In the training phase, each segment $SegEv_j$ of event expression constitutes a minimal unit of an event, from which features for learning the classifier are extracted. In the testing phase, the classes of each system's outputs are predicted against each segment $SegEv_j$.

3.3.2 Features and Classes

In principle, features for learning the classifier for combining outputs of named entity chunkers are represented as a set of pairs of the system indices list $\langle p, \ldots, q \rangle$ and a feature expression F of the named entity:

$$f = \left\{ \begin{array}{c} \langle systems = \langle p, \ldots, q \rangle,\ F \rangle \\ \cdots \\ \langle systems = \langle p', \ldots, q' \rangle,\ F' \rangle \end{array} \right\} \quad (2)$$

In the training phase, any possible feature of this form is extracted from each segment $SegEv_j$ of event expression. The system indices list $\langle p, \ldots, q \rangle$ indicates the list of the systems which output the named entity. A feature expression F of the named entity can be any possible subset of the full feature expression $\{mlength = \cdots, NEtag = \cdots, POS = \cdots\}$, or the set indicating that the system outputs no named entity within the segment.

$$F = \left\{ \begin{array}{l} \text{any subset of } \left\{ mlength = \cdots, \right. \\ \qquad \left. NEtag = \cdots,\ POS = \cdots \right\} \\ \left\{ class_{sys} = \text{"no outputs"} \right\} \end{array} \right.$$

In the training and testing phases, within each segment $SegEv_j$ of event expression, a class is assigned to each system, where each class $class^i_{sys}$ for the i-th system is represented as a list of the classes of the named entities output by the system:

$$class^i_{sys} = \left\{ \begin{array}{ll} +/-, \ \ldots, \ +/- & (i = 1, \ldots, n) \\ \text{"no output"} & \end{array} \right.$$

3.4 Learning Algorithm

We apply a simple decision list learning method to the task of learning a classifier for combining outputs of named entity chunkers[4]. A decision list (Yarowsky, 1994) is a sorted list of decision rules, each of which decides the value of $class$ given some *features* f of an event. Each decision rule in a decision list is sorted in descending order with respect to some preference value, and rules with higher preference values are applied first when applying the decision list to some new test data. In this paper, we simply sort the decision list according to the conditional probability $P(class_i \mid f)$ of the $class_i$ of the i-th system's output given a feature f.

4 Experimental Evaluation

We experimentally evaluate the performance of the proposed system combination method using the IREX workshop's training and test data.

4.1 Comparison of Outputs of Individual Systems

First, Table 3 shows the performance of the individual models described in the section 2.3.2, where trained with the IREX workshop's training data, and tested against the IREX workshop's test data as Ts. The 5-gram model performs the best among those individual models.

Next, assuming that each of the models other than the 5-gram model is combined with the 5-gram model, Table 4 compares the named entities of their outputs. Recall rate of the correct named entities in the union of their outputs, as well as the overlap rate[5] of the over-generated entities against those included in the output of the 5-gram model are shown.

From the Tables 3 and 4, it is clear that the 7-gram and 9-gram models are quite similar to the 5-gram model both in the performance and in the distribution of correct/over-generated named entities. On the other hand, variable length models have distribution of correct/over-generated named entities a lit-

[4] It is quite straightforward to apply any other supervised learning algorithms to this task.

[5] For a model X, *the overlap rate of the over-generated entities against those included in the output of the 5-gram model* is defined as: (# of the intersection of the over-generated entities output by the 5-gram model and those output by the model X)/ (# of the over-generated entities output by the 5-gram model).

Table 3: Performance of Individual Models against Ts (F-measure ($\beta = 1$) (%))

	Features for $M_{\{(-4),-3,3,(4)\}}$		
	All	Lex+POS	Lex
7-gram	80.78	80.81	80.71
9-gram	80.13	80.53	80.53
variable length	45.12	77.02	75.16
5-gram	**81.16**		

Table 4: Difference between 5-gram model and Other Individual Models (Recall of the Union / Overlap Rate of Over-generated Entities) (%)

	Features for $M_{\{(-4),-3,3,(4)\}}$		
	All	Lex+POS	Lex
7-gram	79.8/85.2	79.8/85.2	79.7/91.2
9-gram	79.7/84.7	79.7/86.1	79.5/90.7
variable length	**82.6/27.3**	81.4/63.4	80.4/72.7

tle different from that of the 5-gram model. Variable length models have lower performance mainly because of the difference between the training and testing phases with respect to the modeling of context lengths. Especially, the variable length model with "all" features of $M_{\{-4,-3,3,4\}}$ has much lower performance as well as significantly different distribution of correct/over-generated named entities. This is because character types features are so general that many (erroneous) named entities are over-generated, while sometimes they contribute to finding named entities that are never detected by any of the other models.

4.2 Results of Combining System Outputs

This section reports the results of combining the output of the 5-gram model with that of 7-gram models, 9-gram models, and the variable length models. As the training data sets TrI and TrC, we evaluate the following two assignments (a) and (b), where D_{CRL} denotes the IREX workshop's training data:

(a) TrI: $D_{CRL} - D^{200}_{CRL}$ (200 articles from D_{CRL})
TrC: D^{200}_{CRL}
(b) $TrI = TrC = D_{CRL}$

We use the IREX workshop's test data for Ts. In the assignment (a), TrI and TrC are disjoint, while in the assignment (b), individual named entity chunkers are applied to their own training data, i.e., closed data. The assignment (b) is for the sake of avoiding data sparseness in learning the classifier for combining outputs of two named entity chunkers.

Table 5 shows the peformance in F-measure ($\beta = 1$) for both assignments (a) and (b). For both (a) and (b), "5-gram + variable length (All)" significantly outperforms the 5-gram model, which is the best model among all the individual models without system combination. It is remarkable that models which perform poorly but extract named entities quite different from those of the best performing model can actually help improve the best model by the proposed method. The performance for the assignment (b) is better than that for the assignment (a). This result claims that the training data size should be larger when learning the classifier for combining outputs of two named entity chunkers.

Table 5: Performance of Combining 5-gram model and Other Individual Models (against Ts, F-measure ($\beta = 1$) (%))

(a) $TrI = D_{CRL} - D^{200}_{CRL}$, $TrC = D^{200}_{CRL}$

	Features for $M_{\{(-4),-3,3,(4)\}}$		
	All	Lex+POS	Lex
7-gram	81.54	81.53	80.60
9-gram	81.31	81.26	80.60
variable length	**83.43**	81.55	81.85

(b) $TrI = TrC = D_{CRL}$

	Features for $M_{\{(-4),-3,3,(4)\}}$		
	All	Lex+POS	Lex
7-gram	81.97	81.83	81.58
9-gram	81.53	81.66	81.52
variable length	**84.07**	83.07	82.50

In the Table 6, for the best performing result (i.e., 5-gram + variable length (All)) as well as the constituent individual models (5-gram model and variable length model (All)), we classify the system output according to the number of constituent morphemes of each named entity. In the Table 7, we classify the system output according to the named entity types. The following summarizes several remarkable points of these results: i) the benefit of the system combination is more in the improvement of precision rather than in that of recall. This means that the proposed system combination technique is useful for detecting over-generation of named entity chunkers, ii) the combined outputs of the 5-gram model and the variable length model improve the results of chunking longer named entities quite well compared with shorter named entities. This is the effect of the variable length features of the variable length model.

Table 6: Evaluation Results of Combining System Outputs, per # of constituent morphemes
($TrI = TrC = D_{CRL}$, F-measure ($\beta = 1$) / Recall / Precision (%))

	n Morphemes to 1 Named Entity				
	$n \geq 1$	$n = 1$	$n = 2$	$n = 3$	$n \geq 4$
5-gram	81.16 78.87/83.60	83.60 84.97/82.28	86.94 85.90/88.00	68.42 63.64/73.98	50.59 35.83/86.00
variable length (All)	45.12 51.50/40.15	53.77 38.69/88.14	56.63 71.37/47.93	33.74 57.34/23.91	16.78 40.00/10.62
5-gram + variable length (All)	**84.07** **81.45/86.86**	**85.06** **85.12/84.99**	**88.96** **87.42/90.56**	**75.19** **69.93/81.30**	**65.96** **51.67/91.18**

Table 7: Evaluation Results of Combining System Outputs, per NE type
($TrI = TrC = D_{CRL}$, F-measure ($\beta = 1$) (Recall, Precision) (%))

	ORGANI-ZATION	PER-SON	LOCA-TION	ARTI-FACT	DATE	TIME	MONEY	PER-CENT
5-gram	67.74 (58.45) (80.53)	81.82 (79.88) (83.85)	77.04 (71.91) (82.96)	30.43 (29.17) (31.82)	91.49 (88.85) (94.29)	**93.20** **(88.89)** **(97.96)**	**92.86** **(86.67)** **(100.00)**	**87.18** **(80.95)** **(94.44)**
variable length (All)	35.48 (37.40) (33.75)	48.45 (48.52) (48.38)	38.47 (32.93) (46.26)	5.80 (22.92) (3.32)	78.60 (81.92) (75.53)	56.90 (61.11) (53.23)	60.61 (66.67) (55.56)	**87.18** **(80.95)** **(94.44)**
5-gram + variable length (All)	**72.18** **(62.88)** **(84.70)**	**84.15** **(81.66)** **(86.79)**	**79.58** **(73.61)** **(86.61)**	**38.71** **(37.50)** **(40.00)**	**92.86** **(90.00)** **(95.90)**	**93.20** **(88.89)** **(97.96)**	**92.86** **(86.67)** **(100.00)**	**87.18** **(80.95)** **(94.44)**

5 Conclusion

This paper proposed a method for learning a classifier to combine outputs of more than one Japanese named entity chunkers. Experimental evaluation showed that the proposed method achieved improvement in F-measure over the best known results with an ME model (Uchimoto et al., 2000), when a complementary model extracted named entities quite differently from the best performing model.

References

A. Borthwick, J. Sterling, E. Agichtein, and R. Grishman. 1998. Exploiting diverse knowledge sources via maximum entropy in named entity recognition. In *Proc. 6th Workshop on VLC*, pages 152–160.

E. Brill and J. Wu. 1998. Classifier combination for improved lexical disambiguation. In *Proc. 17th COLING and 36th ACL*, pages 191–195.

S. Della Pietra, V. Della Pietra, and J. Lafferty. 1997. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):380–393.

J. C. Henderson and E. Brill. 1999. Exploiting diversity in natural language processing: Combining parsers. In *Proc. 1999 EMNLP and VLC*, pages 187–194.

J. C. Henderson and E. Brill. 2000. Bagging and boosting a treebank parser. In *Proc. 1st NAACL*, pages 34–41.

IREX Committee, editor. 1999. *Proceedings of the IREX Workshop*. (in Japanese).

M. Sassano and T. Utsuro. 2000. Named entity chunking techniques in supervised learning for Japanese named entity recognition. In *Proceedings of the 18th COLING*, pages 705–711.

M. Sassano, Y. Saito, and K. Matsui. 1997. Japanese morphological analyzer for NLP applications. In *Proc. 3rd Annual Meeting of the Association for Natural Language Processing*, pages 441–444. (in Japanese).

S. Sekine, R. Grishman, and H. Shinnou. 1998. A decision tree method for finding and classifying names in Japanese texts. In *Proc. 6th Workshop on VLC*, pages 148–152.

E. Tjong Kim Sang. 2000. Noun phrase recognition by system combination. In *Proc. 1st NAACL*, pages 50–55.

K. Uchimoto, Q. Ma, M. Murata, H. Ozaku, and H. Isahara. 2000. Named entity extraction based on a maximum entropy model and transformation rules. In *Proc. 38th ACL*, pages 326–335.

H. van Halteren, J. Zavrel, and W. Daelemans. 1998. Improving data driven wordclass tagging by system combination. In *Proc. 17th COLING and 36th ACL*, pages 491–497.

D. H. Wolpert. 1992. Stacked generalization. *Neural Networks*, 5:241–259.

D. Yarowsky. 1994. Decision lists for lexical ambiguity resolution: Application to accent restoration in Spanish and French. In *Proc. 32nd ACL*, pages 88–95.