Inducing Gazetteers for Named Entity Recognition by Large-scale Clustering of Dependency Relations

Jun'ichi Kazama
Japan Advanced Institute of Science and Technology (JAIST),
Asahidai 1-1, Nomi,
Ishikawa, 923-1292 Japan
kazama@jaist.ac.jp

Kentaro Torisawa
National Institute of Information and Communications Technology (NICT),
3-5 Hikaridai, Seika-cho, Soraku-gun,
Kyoto, 619-0289 Japan
torisawa@nict.go.jp

Abstract

We propose using large-scale clustering of dependency relations between verbs and multi-word nouns (MNs) to construct a gazetteer for named entity recognition (NER). Since dependency relations capture the semantics of MNs well, the MN clusters constructed by using dependency relations should serve as a good gazetteer. However, the high level of computational cost has prevented the use of clustering for constructing gazetteers. We parallelized a clustering algorithm based on expectation-maximization (EM) and thus enabled the construction of large-scale MN clusters. We demonstrated with the IREX dataset for the Japanese NER that using the constructed clusters as a gazetteer (*cluster gazetteer*) is a effective way of improving the accuracy of NER. Moreover, we demonstrate that the combination of the cluster gazetteer and a gazetteer extracted from Wikipedia, which is also useful for NER, can further improve the accuracy in several cases.

1 Introduction

Gazetteers, or entity dictionaries, are important for performing named entity recognition (NER) accurately. Since building and maintaining high-quality gazetteers by hand is very expensive, many methods have been proposed for automatic extraction of gazetteers from texts (Riloff and Jones, 1999; Thelen and Riloff, 2002; Etzioni et al., 2005; Shinzato et al., 2006; Talukdar et al., 2006; Nadeau et al., 2006).

Most studies using gazetteers for NER are based on the assumption that a gazetteer is a mapping from a multi-word noun (MN)[1] to named entity categories such as "Tokyo Stock Exchange → {ORGANIZATION}".[2] However, since the correspondence between the labels and the NE categories can be learned by tagging models, a gazetteer will be useful as long as it returns consistent labels even if those returned are not the NE categories. By changing the perspective in such a way, we can explore more broad classes of gazetteers. For example, we can use automatically extracted hyponymy relations (Hearst, 1992; Shinzato and Torisawa, 2004), or automatically induced MN clusters (Rooth et al., 1999; Torisawa, 2001).

For instance, Kazama and Torisawa (2007) used the hyponymy relations extracted from Wikipedia for the English NER, and reported improved accuracies with such a gazetteer.

We focused on the automatically induced clusters of multi-word nouns (MNs) as the source of gazetteers. We call the constructed gazetteers *cluster gazetteers*. In the context of tagging, there are several studies that utilized *word* clusters to prevent the data sparseness problem (Kazama et al., 2001; Miller et al., 2004). However, these methods cannot produce the MN clusters required for constructing gazetteers. In addition, the clustering methods used, such as HMMs and Brown's algorithm (Brown et al., 1992), seem unable to adequately capture the semantics of MNs since they are based only on the information of adjacent words. We utilized richer

[1] We used the term, "multi-word", to emphasize that a gazetteer includes not only one-word expressions but also multi-word expressions.

[2] Although several categories can be associated in general, we assume that only one category is associated.

syntactic/semantic structures, i.e., verb-MN dependencies to make clean MN clusters. Rooth et al. (1999) and Torisawa (2001) showed that the EM-based clustering using verb-MN dependencies can produce semantically clean MN clusters. However, the clustering algorithms, especially the EM-based algorithms, are computationally expensive. Therefore, performing the clustering with a vocabulary that is large enough to cover the many named entities required to improve the accuracy of NER is difficult. We enabled such large-scale clustering by parallelizing the clustering algorithm, and we demonstrate the usefulness of the gazetteer constructed.

We parallelized the algorithm of (Torisawa, 2001) using the Message Passing Interface (MPI), with the prime goal being to distribute parameters and thus enable clustering with a large vocabulary. Applying the parallelized clustering to a large set of dependencies collected from Web documents enabled us to construct gazetteers with up to 500,000 entries and 3,000 classes.

In our experiments, we used the IREX dataset (Sekine and Isahara, 2000) to demonstrate the usefulness of cluster gazetteers. We also compared the cluster gazetteers with the Wikipedia gazetteer constructed by following the method of (Kazama and Torisawa, 2007). The improvement was larger for the cluster gazetteer than for the Wikipedia gazetteer. We also investigated whether these gazetteers improve the accuracies further when they are used in combination. The experimental results indicated that the accuracy improved further in several cases and showed that these gazetteers complement each other.

The paper is organized as follows. In Section 2, we explain the construction of cluster gazetteers and its parallelization, along with a brief explanation of the construction of the Wikipedia gazetteer. In Section 3, we explain how to use these gazetteers as features in an NE tagger. Our experimental results are reported in Section 4.

2 Gazetteer Induction

2.1 Induction by MN Clustering

Assume we have a probabilistic model of a multi-word noun (MN) and its class: $p(n, c) = p(n|c)p(c)$, where $n \in \mathcal{N}$ is an MN and $c \in \mathcal{C}$ is a class. We can use this model to construct a gazetteer in several ways. The method we used in this study constructs a gazetteer: $n \rightarrow \operatorname{argmax}_c p(c|n)$. This computation can be re-written by the Bayes rule as $\operatorname{argmax}_c p(n|c)p(c)$ using $p(n|c)$ and $p(c)$.

Note that we do not exclude non-NEs when we construct the gazetteer. We expect that tagging models (CRFs in our case) can learn an appropriate weight for each gazetteer match regardless of whether it is an NE or not.

2.2 EM-based Clustering using Dependency Relations

To learn $p(n|c)$ and $p(c)$ for Japanese, we use the EM-based clustering method presented by Torisawa (2001). This method assumes a probabilistic model of verb-MN dependencies with hidden semantic classes:[3]

$$p(v, r, n) = \sum_c p(\langle v, r\rangle|c)p(n|c)p(c), \quad (1)$$

where $v \in \mathcal{V}$ is a verb and $n \in \mathcal{N}$ is an MN that depends on verb v with relation r. A relation, r, is represented by Japanese postpositions attached to n. For example, from the following Japanese sentence, we extract the following dependency: $v =$ 飲む (drink), $r =$ を ("wo" postposition), $n =$ ビール (beer).

ビール (beer) を (wo) 飲む (drink) (≈ drink beer)

In the following, we let $vt \equiv \langle v, r\rangle \in \mathcal{VT}$ for the simplicity of explanation.

To be precise, we attach various auxiliary verb suffixes, such as "れる (reru)", which is for passivization, into v, since these greatly change the type of n in the dependent position. In addition, we also treated the MN-MN expressions, "MN_1 の MN_2" (≈ "MN_2 of MN_1"), as dependencies $v = MN_2$, $r =$ の, $n = MN_1$, since these expressions also characterize the dependent MNs well.

Given L training examples of verb-MN dependencies $\{(vt_i, n_i, f_i)\}_{i=1}^{L}$, where f_i is the number of dependency (vt_i, n_i) in a corpus, the EM-based clustering tries to find $p(vt|c)$, $p(n|c)$, and $p(c)$ that maximize the (log)-likelihood of the training examples:

$$LL(p) = \sum_i f_i \log(\sum_c p(vt_i|c)p(n_i|c)p(c)). \quad (2)$$

[3]This formulation is based on the formulation presented in Rooth et al. (1999) for English.

We iteratively update the probabilities using the EM algorithm. For the update procedures used, see Torisawa (2001).

The corpus we used for collecting dependencies was a large set (76 million) of Web documents, that were processed by a dependency parser, KNP (Kurohashi and Kawahara, 2005).[4] From this corpus, we extracted about 380 million dependencies of the form $\{(vt_i, n_i, f_i)\}_i^L$.

2.3 Parallelization for Large-scale Data

The disadvantage of the clustering algorithm described above is the computational costs. The space requirements are $O(|\mathcal{VT}||\mathcal{C}|+|\mathcal{N}||\mathcal{C}|+|\mathcal{C}|)$ for storing the parameters, $p(vt|c)$, $p(n|c)$, and $p(c)$[5], plus $O(L)$ for storing the training examples. The time complexity is mainly $O(L \times |\mathcal{C}| \times I)$, where I is the number of update iterations. The space requirements are the main limiting factor. Assume that a floating-point number consumes 8 bytes. With the setting, $|\mathcal{N}| = 500,000$, $|\mathcal{VT}| = 500,000$, and $|\mathcal{C}| = 3,000$, the algorithm requires more than 44 GB for the parameters and 4 GB of memory for the training examples. A machine with more than 48 GB of memory is not widely available even today.

Therefore, we parallelized the clustering algorithm, to make it suitable for running on a cluster of PCs with a moderate amount of memory (e.g., 8 GB). First, we decided to store the training examples on a file since otherwise each node would need to store all the examples when we use the data splitting described below, and having every node consume 4 GB of memory is memory-consuming. Since the access to the training data is sequential, this does not slow down the execution when we use a buffering technique appropriately.[6]

We then split the matrix for the model parameters, $p(n|c)$ and $p(vt|c)$, along with the class coordinate. That is, each cluster node is responsible for storing only a part of classes $\mathcal{C}_l$, i.e., $1/|P|$ of the parameter matrix, where P is the number of cluster nodes. This data splitting enables linear scalability of memory sizes. However, doing so complicates the update procedure and, in terms of execution speed, may offset the advantage of parallelization because each node needs to receive information about the classes that are not on the node in the inner-most routine of the update procedure.

Algorithm 2.1: Compute $p(c_l|vt_i, n_i)$

$localZ = 0, Z = 0$
for $c_l \in \mathcal{C}_l$ **do**
$\begin{cases} d = p(vt_i|c)p(n_i|c)p(c) \\ p(c_l|vt_i, n_i) = d \\ localZ \ += \ d \end{cases}$
MPI_Allreduce($localZ$, Z, 1, MPI_DOUBLE, MPI_SUM, MPI_COMM_WORLD)
for $c_l \in \mathcal{C}_l$ **do** $p(c_l|vt_i, n_i) \ /= \ Z$

Figure 1: Parallelized inner-most routine of EM clustering algorithm. Each node executes this code in parallel.

The inner-most routine should compute:

$$p(c|vt_i, n_i) = p(vt_i|c)p(n_i|c)p(c)/Z, \qquad (3)$$

for each class c, where $Z = \sum_c p(vt_i|c)p(n_i|c)p(c)$ is a normalizing constant. However, Z cannot be calculated without knowing the results of other cluster nodes. Thus, if we use MPI for parallelization, the parallelized version of this routine should resemble the algorithm shown in Figure 1. This routine first computes $p(vt_i|c_l)p(n_i|c_l)p(c_l)$ for each $c_l \in \mathcal{C}_l$, and stores the sum of these values as $localZ$. The routine uses an MPI function, MPI_Allreduce, to sum up $localZ$ of the all cluster nodes and to set Z with the resulting sum. We can compute $p(c_l|vt_i, n_i)$ by using this Z to normalize the value. Although the above is the essence of our parallelization, invoking MPI_Allreduce in the inner-most loop is very expensive because the communication setup is not so cheap. Therefore, our implementation calculates $p(c_l|vt_i, n_i)$ in batches of B examples and calls MPI_Allreduce at every B examples.[7] We used a value of $B = 4,096$ in this study.

By using this parallelization, we successfully performed the clustering with $|\mathcal{N}| = 500,000$, $|\mathcal{VT}| = 500,000$, $|\mathcal{C}| = 3,000$, and $I = 150$, on 8 cluster nodes with a 2.6 GHz Opteron processor and 8 GB of memory. This clustering took about a week. To our knowledge, no one else has performed EM-based clustering of this type on this scale. The resulting MN clusters are shown in Figure 2. In terms of speed, our experiments are still at a preliminary

[4]**Acknowledgements:** This corpus was provided by Dr. Daisuke Kawahara of NICT.

[5]To be precise, we need two copies of these.

[6]Each node has a copy of the training data on a local disk.

[7]MPI_Allreduce can also take array arguments and apply the operation to each element of the array in one call.

Class 791	Class 2760
ウィン ダ ム (WINDOM)	マリン/スタジアム (Chiba Marine Stadium [abb.])
カ ム リ (CAMRY)	大阪/ドーム (Osaka Dome)
ディア マ ン テ (DIAMANTE)	ナ ゴ/ド (Nagoya Dome [abb.])
オ デッセ イ (ODYSSEY)	福岡/ドーム (Fukuoka Dome)
イ ン ス パ イ ア (INSPIRE)	大阪/球場 (Osaka Stadium)
ス イ フ ト (SWIFT)	ハマ/スタ (Yokohama Stadium [abb.])

Figure 2: Clean MN clusters with named entity entries (Left: car brand names. Right: stadium names). Names are sorted on the basis of $p(c|n)$. Stadium names are examples of multi-word nouns (word boundaries are indicated by "/") and also include abbreviated expressions (marked by [abb.]) .

stage. We have observed 5 times faster execution, when using 8 cluster nodes with a relatively small setting, $|\mathcal{N}| = |\mathcal{VT}| = 50,000$, $|\mathcal{C}| = 2,000$.

2.4 Induction from Wikipedia

Defining sentences in a dictionary or an encyclopedia have long been used as a source of hyponymy relations (Tsurumaru et al., 1991; Herbelot and Copestake, 2006).

Kazama and Torisawa (2007) extracted hyponymy relations from the first sentences (i.e., defining sentences) of Wikipedia articles and then used them as a gazetteer for NER. We used this method to construct the Wikipedia gazetteer.

The method described by Kazama and Torisawa (2007) is to first extract the first (base) noun phrase after the first "is", "was", "are", or "were" in the first sentence of a Wikipedia article. The last word in the noun phase is then extracted and becomes the hypernym of the entity described by the article. For example, from the following defining sentence, it extracts "guitarist" as the hypernym for "Jimi Hendrix".

> Jimi Hendrix (November 27, 1942) was an American guitarist, singer and songwriter.

The second noun phrase is used when the first noun phrase ends with "one", "kind", "sort", or "type", or it ended with "name" followed by "of". This rule is for treating expressions like "... is one of the landlocked countries." By applying this method of extraction to all the articles in Wikipedia, we construct a gazetteer that maps an MN (a title of a Wikipedia article) to its hypernym.[8] When the hypernym extraction failed, a special hypernym symbol, e.g., "UNK", was used.

	# instances
page titles processed	550,832
articles found (found by redirection)	547,779 (189,222)
first sentences found	545,577
hypernyms extracted	482,599

Table 1: Wikipedia gazetteer extraction

We modified this method for Japanese. After preprocessing the first sentence of an article using a morphological analyzer, MeCab[9], we extracted the *last* noun after the appearance of Japanese postposition "は (wa)" (≈ "is"). As in the English case, we also refrained from extracting expressions corresponding to "one of" and so on.

From the Japanese Wikipedia entries of April 10, 2007, we extracted 550,832 gazetteer entries (482,599 entries have hypernyms other than UNK). Various statistics for this extraction are shown in Table 1. The number of distinct hypernyms in the gazetteer was 12,786. Although this Wikipedia gazetteer is much smaller than the English version used by Kazama and Torisawa (2007) that has over 2,000,000 entries, it is the largest gazetteer that can be freely used for Japanese NER. Our experimental results show that this Wikipedia gazetteer can be used to improve the accuracy of Japanese NER.

3 Using Gazetteers as Features of NER

Since Japanese has no spaces between words, there are several choices for the token unit used in NER. Asahara and Motsumoto (2003) proposed using characters instead of morphemes as the unit to alleviate the effect of segmentation errors in morphological analysis and we also used their character-based method. The NER task is then treated as a tagging task, which assigns IOB tags to each character in a sentence.[10] We use Conditional Random Fields (CRFs) (Lafferty et al., 2001) to perform this tagging.

The information of a gazetteer is incorporated

[8] They handled "redirections" as well by following redirection links and extracting a hypernym from the article reached.

[9] `http://mecab.sourceforge.net`

[10] Precisely, we use IOB2 tags.

ch	に	ソ	ニ	ー	が	開	発	···
match	O	B	I	I	O	O	O	···
(w/ class)	O	B-会社	I-会社	I-会社	O	O	O	···

Figure 3: Gazetteer features for Japanese NER. Here, 'ソニー" means "SONY", "会社" means "company", and "開発" means "to develop".

as features in a CRF-based NE tagger. We follow the method used by Kazama and Torisawa (2007), which encodes the matching with a gazetteer entity using IOB tags, with the modification for Japanese. They describe using two types of gazetteer features. The first is a matching-only feature, which uses bare IOB tags to encode only matching information. The second uses IOB tags that are augmented with classes (e.g., B-country and I-country).[11] When there are several possibilities for making a match, the left-most longest match is selected. The small differences from their work are: (1) We used characters as the unit as we described above, (2) While Kazama and Torisawa (2007) checked only the word sequences that start with a capitalized word and thus exploited the characteristics of English language, we checked the matching at every character, (3) We used a TRIE to make the look-up efficient.

The output of gazetteer features for Japanese NER are thus as those shown in Figure 3. These annotated IOB tags can be used in the same way as other features in a CRF tagger.

4 Experiments

4.1 Data

We used the CRL NE dataset provided in the IREX competition (Sekine and Isahara, 2000). In the dataset, 1,174 newspaper articles are annotated with 8 NE categories: ARTIFACT, DATE, LOCATION, MONEY, ORGANIZATION, PERCENT, PERSON, and TIME.[12] We converted the data into the CoNLL 2003 format, i.e., each row corresponds to a character in this case. We obtained 11,892 sentences[13] with 18,677 named entities. We split this data into the training set (9,000 sentences), the development set (1,446 sentences), and the testing set (1,446 sentences).

[11]Here, we call the value returned by a gazetteer a "class". Features are not output when the returned class is UNK in the case of the Wikipedia gazetteer. We did not observe any significant change if we also used UNK.

[12]We ignored OPTIONAL category.

[13]This number includes the number of -DOCSTART- tokens in CoNLL 2003 format.

Name	Description
ch	character itself
ct	character type: uppercase alphabet, lowercase alphabet, katakana, hiragana, Chinese characters, numbers, numbers in Chinese characters, and spaces
m_mo	bare IOB tag indicating boundaries of morphemes
m_mm	IOB tag augmented by morpheme string, indicating boundaries and morphemes
m_mp	IOB tag augmented by morpheme type, indicating boundaries and morpheme types (POSs)
bm	bare IOB tag indicating "bunsetsu" boundaries (Bunsetsu is a basic unit in Japanese and usually contains content words followed by function words such as postpositions)
bi	bunsetsu-inner feature. See (Nakano and Hirai, 2004).
bp	adjacent-bunsetsu feature. See (Nakano and Hirai, 2004).
bh	head-of-bunsetsu features. See (Nakano and Hirai, 2004).

Table 2: Atomic features used in baseline model.

4.2 Baseline Model

We extracted the atomic features listed in Table 2 at each character for our baseline model. Though there may be slight differences, these features are based on the standard ones proposed and used in previous studies on Japanese NER such as those by Asahara and Motsumoto (2003), Nakano and Hirai (2004), and Yamada (2007). We used MeCab as a morphological analyzer and CaboCha[14] (Kudo and Matsumoto, 2002) as the dependency parser to find the boundaries of the bunsetsu. We generated the node and the edge features of a CRF model as described in Table 3 using these atomic features.

4.3 Training

To train CRF models, we used Taku Kudo's CRF++ (ver. 0.44) [15] with some modifications.[16] We

[14]http://chasen.org/~taku/software/CaboCha

[15]http://chasen.org/~taku/software/CRF++

[16]We implemented scaling, which is similar to that for HMMs (Rabiner, 1989), in the forward-backward phase and replaced the optimization module in the original package with the

Node features:
$\{"", \mathrm{x}_{-2}, \mathrm{x}_{-1}, \mathrm{x}_0, \mathrm{x}_{+1}, \mathrm{x}_{+2}\} \times y_0$ where x = ch, ct, m_mm, m_mo, m_mp, bi, bp, and bh
Edge features:
$\{"", \mathrm{x}_{-1}, \mathrm{x}_0, \mathrm{x}_{+1}\} \times y_{-1} \times y_0$ where x = ch, ct, and m_mp
Bigram node features:
$\{\mathrm{x}_{-2}\mathrm{x}_{-1}, \mathrm{x}_{-1}\mathrm{x}_0, \mathrm{x}_0\mathrm{x}_{+1}\} \times y_0$ x = ch, ct, m_mo, m_mp, bm, bi, bp, and bh

Table 3: Baseline features. Value of node feature is determined from current tag, y_0, and surface feature (combination of atomic features in Table 2). Value of edge feature is determined by previous tag, y_{-1}, current tag, y_0, and surface feature. Subscripts indicate relative position from current character.

used Gaussian regularization to prevent overfitting. The parameter of the Gaussian, σ^2, was tuned using the development set. We tested 10 points: $\{0.64, 1.28, 2.56, 5.12, \ldots, 163.84, 327.68\}$. We stopped training when the relative change in the log-likelihood became less than a pre-defined threshold, 0.0001. Throughout the experiments, we omitted the features whose surface part described in Table 3 occurred less than twice in the training corpus.

4.4 Effect of Gazetteer Features

We investigated the effect of the cluster gazetteer described in Section 2.1 and the Wikipedia gazetteer described in Section 2.4, by adding each gazetteer to the baseline model. We added the matching-only and the class-augmented features, and we generated the node and the edge features in Table 3.[17] For the cluster gazetteer, we made several gazetteers that had different vocabulary sizes and numbers of classes. The number of clustering iterations was 150 and the initial parameters were set randomly with a Dirichlet distribution ($\alpha_i = 1.0$).

The statistics of each gazetteer are summarized in Table 4. The number of entries in a gazetteer is given by "# entries", and "# matches" is the number of matches that were output for the training set. We define "# e-matches" as the number of matches that also match a boundary of a named entity in the training set, and "# optimal" as the optimal number of "# e-matches" that can be achieved when we know the oracle of entity boundaries. Note that this cannot be realized because our matching uses the left-most longest heuristics. We define "pre." as the precision of the output matches (i.e., # e-matches/# matches), and "rec." as the recall (i.e., # e-matches/# NEs). Here, # NEs $= 14,056$. Finally, "opt." is the optimal recall (i.e., # optimal/# NEs). "# classes" is the number of distinct classes in a gazetteer, and "# used" is the number of classes that were output for the training set. Gazetteers are as follows: "wikip(m)" is the Wikipedia gazetteer (matching only), and "wikip(c)" is the Wikipedia gazetteer (with class-augmentation). A cluster gazetteer, which is constructed by the clustering with $|\mathcal{N}| = |\mathcal{VT}| = X \times 1,000$ and $|\mathcal{C}| = Y \times 1,000$, is indicated by "c$X$k-$Y$k". Note that "# entries" is slightly smaller than the vocabulary size since we removed some duplications during the conversion to a TRIE.

These gazetteers cover 40 - 50% of the named entities, and the cluster gazetteers have relatively wider coverage than the Wikipedia gazetteer has. The precisions are very low because there are many erroneous matches, e.g., with a entries for a hiragana character.[18] Although this seems to be a serious problem, removing such one-character entries does not affect the accuracy, and in fact, makes it worsen slightly. We think this shows one of the strengths of machine learning methods such as CRFs. We can also see that our current matching method is not an optimal one. For example, 16% of the matches were lost as a result of using our left-most longest heuristics for the case of the c500k-2k gazetteer.

A comparison of the effect of these gazetteers is shown in Table 5. The performance is measured by the F-measure. First, the Wikipedia gazetteer improved the accuracy as expected, i.e., it reproduced the result of Kazama and Torisawa (2007) for Japanese NER. The improvement for the testing set was 1.08 points. Second, all the tested cluster gazetteers improved the accuracy. The largest improvement was 1.55 points with the c300k-3k gazetteer. This was larger than that of the Wikipedia gazetteer. The results for c300k-Yk gazetteers show a peak of the improvement at some number of clusters. In this case, $|\mathcal{C}| = 3,000$ achieved the best improvement. The results of cXk-2k gazetteers in-

LMVM optimizer of TAO (version 1.9) (Benson et al., 2007)

[17] Bigram node features were not used for gazetteer features.

[18] Wikipedia contains articles explaining each hiragana character, e.g., "あ is a hiragana character".

Name	# entries	# matches	# e-matches	# optimal	pre. (%)	rec. (%)	opt. rec. (%)	# classes	# used
wikip(m)	550,054	225,607	6,804	7,602	3.02	48.4	54.1	N/A	N/A
wikip(c)	550,054	189,029	5,441	6,064	2.88	38.7	43.1	12,786	1,708
c100k-2k	99,671	193,897	6,822	8,233	3.52	48.5	58.6	2,000	1,910
c300k-2k	295,695	178,220	7,377	9,436	4.14	52.5	67.1	2,000	1,973
c300k-1k	↑	↑	↑	↑	↑	↑	↑	1,000	982
c300k-3k	↑	↑	↑	↑	↑	↑	↑	3,000	2,848
c300k-4k	↑	↑	↑	↑	↑	↑	↑	4,000	3,681
c500k-2k	497,101	174,482	7,470	9,798	4.28	53.1	69.7	2,000	1,951
c500k-3k	↑	↑	↑	↑	↑	↑	↑	3,000	2,854

Table 4: Statistics of various gazetteers.

Model	F (dev.)	F (test.)	best σ^2
baseline	87.23	87.42	20.48
+wikip	**87.60**	**88.50**	2.56
+c300k-1k	88.74	87.98	40.96
+c300k-2k	88.75	88.01	163.84
+c300k-3k	**89.12**	**88.97**	20.48
+c300k-4k	88.99	88.40	327.68
+c100k-2k	88.15	88.06	20.48
+c500k-2k	88.80	88.12	40.96
+c500k-3k	88.75	88.03	20.48

Table 5: Comparison of gazetteer features.

Model	F (dev.)	F (test.)	best σ^2
+wikip+c300k-1k	88.65	*89.32	0.64
+wikip+c300k-2k	*89.22	*89.13	10.24
+wikip+c300k-3k	88.69	*89.62	40.96
+wikip+c300k-4k	88.67	*89.19	40.96
+wikip+c500k-2k	***89.26**	***89.19**	2.56
+wikip+c500k-3k	*88.80	*88.60	10.24

Table 6: Effect of combination. Figures with * mean that accuracy was improved by combining gazetteers.

dicate that the larger a gazetteer is, the larger the improvement. However, the accuracies of the c300k-3k and c500k-3k gazetteers seem to contradict this tendency. It might be caused by the accidental low quality of the clustering that results from random initialization. We need to investigate this further.

4.5 Effect of Combining the Cluster and the Wikipedia Gazetteers

We have observed that using the cluster gazetteer and the Wikipedia one improves the accuracy of Japanese NER. The next question is whether these gazetteers improve the accuracy further when they are used together. The accuracies of models that use the Wikipedia gazetteer and one of the cluster gazetteers at the same time are shown in Table 6. The accuracy was improved in most cases. However, there were some cases where the accuracy for the development set was degraded. Therefore, we should state at this point that while the benefit of combining these gazetteers is not consistent in a strict sense, it seems to exist. The best performance, F = 89.26 (dev.) / 89.19 (test.), was achieved when we combined the Wikipedia gazetteer and the cluster gazetteer, c500k-2k. This means that there was a 1.77-point improvement from the baseline for the testing set.

Model	F
(Asahara and Motsumoto, 2003)	87.21
(Nakano and Hirai, 2004)	89.03
(Yamada, 2007)	88.33
(Sasano and Kurohashi, 2008)	89.40
proposed (baseline)	87.62
proposed (+wikip)	88.14
proposed (+c300k-3k)	88.45
proposed (+c500k-2k)	88.41
proposed (+wikip+c300k-3k)	**88.93**
proposed (+wikip+c500k-2k)	88.71

Table 7: Comparison with previous studies

5 Comparison with Previous Studies

Since many previous studies on Japanese NER used 5-fold cross validation for the IREX dataset, we also performed it for some our models that had the best σ^2 found in the previous experiments. The results are listed in Table 7 with references to the results of recent studies. These results not only reconfirmed the effects of the gazetteer features shown in the previous experiments, but they also showed that our best model is comparable to the state-of-the-art models. The system recently proposed by Sasano and Kurohashi (2008) is currently the best system for the IREX dataset. It uses many structural features that are not used in our model. Incorporating

such features might improve our model further.

6 Related Work and Discussion

There are several studies that used automatically extracted gazetteers for NER (Shinzato et al., 2006; Talukdar et al., 2006; Nadeau et al., 2006; Kazama and Torisawa, 2007). Most of the methods (Shinzato et al., 2006; Talukdar et al., 2006; Nadeau et al., 2006) are oriented at the NE category. They extracted a gazetteer for each NE category and utilized it in a NE tagger. On the other hand, Kazama and Torisawa (2007) extracted hyponymy relations, which are independent of the NE categories, from Wikipedia and utilized it as a gazetteer. The effectiveness of this method was demonstrated for Japanese NER as well by this study.

Inducing features for taggers by clustering has been tried by several researchers (Kazama et al., 2001; Miller et al., 2004). They constructed word clusters by using HMMs or Brown's clustering algorithm (Brown et al., 1992), which utilize only information from neighboring words. This study, on the other hand, utilized MN clustering based on verb-MN dependencies (Rooth et al., 1999; Torisawa, 2001). We showed that gazetteers created by using such richer semantic/syntactic structures improves the accuracy for NER.

The size of the gazetteers is also a novel point of this study. The previous studies, with the exception of Kazama and Torisawa (2007), used smaller gazetteers than ours. Shinzato et al. (2006) constructed gazetteers with about 100,000 entries in total for the "restaurant" domain; Talukdar et al. (2006) used gazetteers with about 120,000 entries in total, and Nadeau et al. (2006) used gazetteers with about 85,000 entries in total. By parallelizing the clustering algorithm, we successfully constructed a cluster gazetteer with up to 500,000 entries from a large amount of dependency relations in Web documents. To our knowledge, no one else has performed this type of clustering on such a large scale. Wikipedia also produced a large gazetteer of more than 550,000 entries. However, comparing these gazetteers and ours precisely is difficult at this point because the detailed information such as the precision and the recall of these gazetteers were not reported.[19] Recently, Inui et al. (2007) investigated the relation between the size and the quality of a gazetteer and its effect. We think this is one of the important directions of future research.

Parallelization has recently regained attention in the machine learning community because of the need for learning from very large sets of data. Chu et al. (2006) presented the MapReduce framework for a wide range of machine learning algorithms, including the EM algorithm. Newman et al. (2007) presented parallelized Latent Dirichlet Allocation (LDA). However, these studies focus on the distribution of the training examples and relevant computation, and ignore the need that we found for the distribution of model parameters. The exception, which we noticed recently, is a study by Wolfe et al. (2007), which describes how each node stores only those parameters relevant to the training data on each node. However, some parameters need to be duplicated and thus their method is less efficient than ours in terms of memory usage.

We used the left-most longest heuristics to find the matching gazetteer entries. However, as shown in Table 4 this is not an optimal method. We need more sophisticated matching methods that can handle multiple matching possibilities. Using models such as Semi-Markov CRFs (Sarawagi and Cohen, 2004), which handle the features on overlapping regions, is one possible direction. However, even if we utilize the current gazetteers optimally, the coverage is upper bounded at 70%. To cover most of the named entities in the data, we need much larger gazetteers. A straightforward approach is to increase the number of Web documents used for the MN clustering and to use larger vocabularies.

7 Conclusion

We demonstrated that a gazetteer obtained by clustering verb-MN dependencies is a useful feature for a Japanese NER. In addition, we demonstrated that using the cluster gazetteer and the gazetteer extracted from Wikipedia (also shown to be useful) can together further improves the accuracy in several cases. Future work will be to refine the matching method and to construct even larger gazetteers.

[19] Shinzato et al. (2006) reported some useful statistics about their gazetteers.

References

M. Asahara and Y. Motsumoto. 2003. Japanese named entity extraction with redundant morphological analysis.

S. Benson, L. C. McInnes, J. Moré, T. Munson, and J. Sarich. 2007. TAO user manual (revision 1.9). Technical Report ANL/MCS-TM-242, Mathematics and Computer Science Division, Argonne National Laboratory. http://www.mcs.anl.gov/tao.

P. F. Brown, V. J. Della Pietra, P. V. deSouza, J. C. Lai, and R. L. Mercer. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479.

C.-T. Chu, S. K. Kim, Y.-A. Lin, Y. Yu, G. Bradski, A. Y. Ng, and K. Olukotun. 2006. Map-reduce for machine learning on multicore. In *NIPS 2006*.

O. Etzioni, M. Cafarella, D. Downey, A. M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates. 2005. Unsupervised named-entity extraction from the Web – an experimental study. *Artificial Intelligence Journal*.

M. A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proc. of the 14th International Conference on Computational Linguistics*, pages 539–545.

A. Herbelot and A. Copestake. 2006. Acquiring ontological relationships from Wikipedia using RMRS. In *Workshop on Web Content Mining with Human Language Technologies ISWC06*.

T. Inui, K. Murakami, T. Hashimoto, K. Utsumi, and M. Ishikawa. 2007. A study on using gazetteers for organization name recognition. In *IPSJ SIG Technical Report 2007-NL-182 (in Japanese)*.

J. Kazama and K. Torisawa. 2007. Exploiting Wikipedia as external knowledge for named entity recognition. In *EMNLP-CoNLL 2007*.

J. Kazama, Y. Miyao, and J. Tsujii. 2001. A maximum entropy tagger with unsupervised hidden Markov models. In *NLPRS 2001*.

T. Kudo and Y. Matsumoto. 2002. Japanese dependency analysis using cascaded chunking. In *CoNLL 2002*.

S. Kurohashi and D. Kawahara. 2005. KNP (Kurohashi-Nagao parser) 2.0 users manual.

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML 2001*.

S. Miller, J. Guinness, and A. Zamanian. 2004. Name tagging with word clusters and discriminative training. In *HLT-NAACL04*.

D. Nadeau, Peter D. Turney, and Stan Matwin. 2006. Unsupervised named-entity recognition: Generating gazetteers and resolving ambiguity. In *19th Canadian Conference on Artificial Intelligence*.

K. Nakano and Y. Hirai. 2004. Japanese named entity extraction with bunsetsu features. *IPSJ Journal (in Japanese)*.

D. Newman, A. Asuncion, P. Smyth, and M. Welling. 2007. Distributed inference for latent dirichlet allocation. In *NIPS 2007*.

L. R. Rabiner. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

E. Riloff and R. Jones. 1999. Learning dictionaries for information extraction by multi-level bootstrapping. In *16th National Conference on Artificial Intelligence (AAAI-99)*.

M. Rooth, S. Riezler, D. Presher, G. Carroll, and F. Beil. 1999. Inducing a semantically annotated lexicon via EM-based clustering.

S. Sarawagi and W. W. Cohen. 2004. Semi-Markov random fields for information extraction. In *NIPS 2004*.

R. Sasano and S. Kurohashi. 2008. Japanese named entity recognition using structural natural language processing. In *IJCNLP 2008*.

S. Sekine and H. Isahara. 2000. IREX: IR and IE evaluation project in Japanese. In *IREX 2000*.

K. Shinzato and K. Torisawa. 2004. Acquiring hyponymy relations from Web documents. In *HLT-NAACL 2004*.

K. Shinzato, S. Sekine, N. Yoshinaga, and K. Torisawa. 2006. Constructing dictionaries for named entity recognition on specific domains from the Web. In *Web Content Mining with Human Language Technologies Workshop on the 5th International Semantic Web*.

P. P. Talukdar, T. Brants, M. Liberman, and F. Pereira. 2006. A context pattern induction method for named entity extraction. In *CoNLL 2006*.

M. Thelen and E. Riloff. 2002. A bootstrapping method for learning semantic lexicons using extraction pattern context. In *EMNLP 2002*.

K. Torisawa. 2001. An unsupervised method for canonicalization of Japanese postpositions. In *NLPRS 2001*.

H. Tsurumaru, K. Takeshita, K. Iami, T. Yanagawa, and S. Yoshida. 1991. An approach to thesaurus construction from Japanese language dictionary. In *IPSJ SIG Notes Natural Language vol.83-16, (in Japanese)*.

J. Wolfe, A. Haghighi, and D. Klein. 2007. Fully distributed EM for very large datasets. In *NIPS Workshop on Efficient Machine Learning*.

H. Yamada. 2007. Shift-reduce chunking for Japanese named entity extraction. In *ISPJ SIG Technical Report 2007-NL-179*.