International Workshop on Lexically Driven Information Extraction ~~EDIE final~~ 1997

Towards the Application of Word Sense Disambiguation to Information Extraction

Atsushi Fujii, Kentaro Inui, Timothy Baldwin, Takenobu Tokunaga
and Hozumi Tanaka

Department of Computer Science, Tokyo Institute of Technology
2-12-1 Ookayama Meguroku Tokyo 152, JAPAN
{fujii,inui,tim,take,tanaka}@cs.titech.ac.jp

Abstract

Word sense disambiguation is a crucial task in many NLP applications, including information extraction. More recent corpus-based word sense disambiguation techniques generally use a database containing supervised examples (annotated with correct word senses), and therefore constructing a database of a practical size involves a considerable overhead for manual sense disambiguation ("overhead for supervision"). In addition, the time complexity of searching a large-sized database poses a considerable problem ("overhead for search"). To counter these problems, we previously proposed a method which selectively samples a smaller-sized informative subset from a given example set, to use in word sense disambiguation. Our method is characterized by its reliance on the notion of "training utility": the degree to which each example is informative for future example sampling when used for the training of the system. The system progressively collects examples by selecting those with greatest utility. In this paper, we extensively compare our method with other existing sampling methods by way of experiments on about 3000 sentences. The result of the experiments showed that our method reduced both the overhead for supervision and the overhead for search to a larger degree than other methods, without degenerating the performance of the system.

1 Introduction

Word sense disambiguation is a crucial task in many NLP applications, such as machine translation [2], parsing [20, 22] and text retrieval [16, 33]. In this paper, we suggest that successful word sense disambiguation is expected to improve the accuracy of information extraction (IE) tasks. In typical IE systems[1], a natural language input undergoes morphological analysis, syntactic analysis, semantic analysis, and discourse analysis to generate templated information. Semantic analysis transforms parse trees to semantic predicate-argument structures, that is, each verb along with its complement(s) in the parse tree is assigned to one of a number of predefined structures. Most proposed systems assume that each verb is *uniquely* associated with a given structure because verbs are expected to have consistent meaning (sense) over a well-defined domain. This may well be true for the limited domain levels targeted in proposed systems. However, by way of diversifying domains, disambiguation of verb senses will become integral to semantic analysis, as most verbs will be associated with multiple senses, i.e. multiple structure candidates. One may argue that selectional restriction (generally carried out manually) described for each argument slot could facilitate the correct selection of structure candidates. However, hand-encoding rules entail considerable human effort, besides which rule-based disambiguation is prone to failure given exceptional cases or unmodeled lexemes.

Given the growing utilization of machine readable texts, word sense disambiguation techniques have been variously proposed in corpus-based approaches. Unlike rule-based approaches (some of which are reviewed, for example, by Hirst [13]), corpus-based approaches release us from the task of generalizing observed phenomena to make a set of rules. In this paper we focus on disambiguation of verb senses

[1] As have been variously proposed at Message Understanding Conferences (MUCs).

based on such an approach, or more precisely an example-based approach. As with most example-based systems [10, 17, 19, 31], our system uses an example database (database, hereafter) which contains example sentences associated with each verb sense. Given an input sentence containing a sense ambiguous verb, the system then chooses the most plausible verb sense from predefined candidates. In this process, the system computes a scored similarity between the input and examples in the database, and choses the verb sense that is related to the example maximizing the score. To realize this, we have to manually disambiguate sense ambiguous verbs appearing in examples, prior to their use by the system. We shall call these examples "supervised examples". In order to apply this technique to IE systems, the following problems have to be taken into account[2]:

- given human resource limitations, it is not reasonable to supervise every example in large corpora ("overhead for supervision"),
- given the fact that example-based systems, including our system, search the database for the most similar examples with regard to the input, the computational cost becomes prohibitive if one works with a very large database size ("overhead for search").

The former problem is also crucial when one tries to customize an IE system to several distinct domains. To counter these problems, we proposed a novel method of selecting a small number of optimally informative examples ("samples") from given corpora, and demonstrated its effectivity in comparison with a random sampling method [9]. In this paper, we describe a more extensive evaluation, in which we compare our sampling method with other existing uncertainty sampling [18] and committee-based sampling [7] methods.

Our example sampling method, based on the utility maximization principle, decides on the preference for including a given example in the database. This decision procedure is usually called *selective sampling* [5]. The overall control flow of selective sampling systems can be depicted as in figure 1, where "system" refers to our verb sense disambiguation system, and "examples" refers to an unsupervised example set. The sampling process basically cycles between the word sense disambiguation (WSD) and training phases. During the WSD phase, the system generates an interpretation for each sense ambiguous verb contained in the input example ("WSD outputs"). This phase is equivalent to normal word sense disambiguation execution. During the training phase, the system selects samples for training from the previously produced outputs. During this phase, a human expert supervises samples, that is, provides the correct interpretation for the verbs appearing in the samples. Thereafter, samples are simply contained in the database without any computational overhead, meaning that the system can be trained on the remaining examples ("residue") for the next iteration. Iterating these two phases, the system progressively enhances the database. Note that the selective sampling procedure gives us an optimally informative database of a given size irrespective of the stage at which processing is terminated.

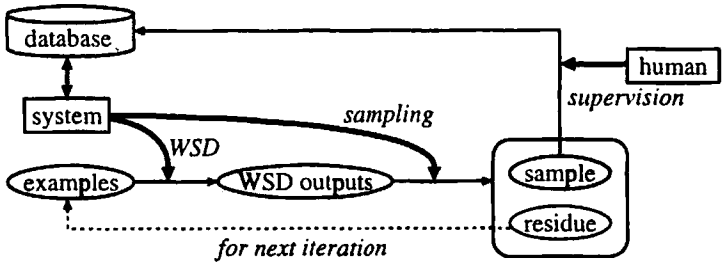

Figure 1: Flow of control of the example sampling system

With respect to the problem of overhead for search, possible solutions would include the generalization of similar examples [14, 24] or the reconstruction of the database using a small portion of useful instances

[2]Note that these problems are associated with corpus-based approaches in general, and have been identified by a number of researchers [7, 18, 30, 34].

selected from a given supervised example set [1, 28]. However, such approaches imply a significant overhead for supervision of each example prior to the system's execution. This shortcoming is precisely what our approach aims to avoid: reduction of the overhead for supervision as well as the overhead for search.

Section 2 briefly describes the basis of our verb sense disambiguation system. Section 3 then elaborates on our example sampling method, and section 4 compares our method with other proposed sampling methods by way of experiments. Discussion is added in section 5, followed by the conclusion.

2 Example-based verb sense disambiguation

Our system, which disambiguates Japanese verbs[3], uses a database containing examples of collocations for each verb sense and its associated case frame(s). Figure 2 shows a fragment of the database associated with the Japanese verb *tsukau*, some of which senses are "to employ", "to operate" and "to spend". The database specifies the case frame(s) associated with each verb sense. In Japanese, a complement of a verb consists of a noun phrase (case filler) and its case marker suffix, for example *ga* (nominative), *ni* (dative) or *wo* (accusative). The database lists several case filler examples for each case. Given an input, the system identifies the verb sense on the basis of the similarity between the input and examples for each verb sense contained in the database. Let us take the following input:

enjinia ga	*fakkusu wo*	*tsukau*
(engineer-NOM)	(facsimile-ACC)	(?)

In this example, one may consider *enjinia* ("engineer") and *fakkusu* ("facsimile") to be semantically similar to *gakusei* ("student") and *konpyuutaa* ("computer"), respectively, from the "to operate" sense of *tsukau*. As a result, *tsukau* is interpreted as "to operate"[4]. To formalize this notion, the system computes the plausibility score for each verb sense candidate, and chooses the sense that maximizes the score. The score is computed by considering the weighted average of the similarity of the input case fillers with respect to each of the corresponding example case fillers listed in the database for the sense under evaluation. Formally, this is expressed by equation (1), where $Score(s)$ is the score for verb sense s. n_C denotes the case filler for case c, and $\mathcal{E}_{s,c}$ denotes a set of case filler examples for each case c of sense s (for example, $\mathcal{E} = \{kare, kigyou\}$ for the *ga* case in the "to employ" sense in figure 2). $sim(n_C, e)$ stands for the similarity between n_C and an example case filler e.

$$Score(s) = \sum_c CCD(c) \cdot \max_{e \in \mathcal{E}_{s,c}} sim(n_C, e) \quad (1)$$

$CCD(c)$ expresses the weight factor of case c using the notion of case contribution to verb sense disambiguation (CCD) proposed by Fujii et al [10]. Intuitively, the CCD of a case becomes greater when example sets of the case fillers are disjunctive over different verb senses. In the case fillers of figure 2, for example, $CCD(ACC)$ is greater than $CCD(NOM)$ (see Fujii et al's paper for details).

Here, let us discuss the overhead for parsing required to identify case fillers. A number of IE systems employ partial parsing to identify only cases more highly proximal to verbs, instead of full parsing. Similarly, our word sense disambiguation method does not require full parsing because our preliminary experiments have shown that (a) cases more highly proximal to a target verb tend to have greater CCD, and (b) we can maintain the performance of word sense disambiguation relying solely on such cases with greatest CCD.

In regard to the computation of the similarity between two different case fillers ($sim(n_C, e)$ in equation (1)), there are two alternative approaches. One approach uses semantic resources, that is, handcrafted thesauri (such as the Roget's thesaurus [3] or WordNet [21] in the case of English, and *Bunruigoihyo* [23] or EDR [6] in the case of Japanese), based on the intuitively feasible assumption that words located near each other within the structure of a thesaurus have similar meaning. Therefore, the similarity between two given words is represented by the length of the path between them in the thesaurus

[3]Our sampling method is theoretically applicable to other languages.

[4]Unlike the automatic acquisition of word sense definitions [11, 32], the task of the system is to identify the best matched category with a given input, from ***predefined*** candidates.

kare (he) *kigyou* (company)	*ga*	*kikaku* (project)	*ni*	*jyuugyouin* (employee) *sotsugyousei* (graduate)	*wo*	*tsukau* (to employ)
kanojo (she) *gakusei* (student)	*ga*	*shigoto* (work) *kenkyuu* (research)	*ni*	*konpyuutaa* (computer) *kikai* (machine)	*wo*	*tsukau* (to operate)
kare (he) *seifu* (government)	*ga*	*kuruma* (car) *fukushi* (welfare)	*ni*	*nenryou* (fuel) *shigen* (resource) *zeikin* (tax)	*wo*	*tsukau* (to spend)

Figure 2: A fragment of the database associated with the Japanese verb *tsukau*

structure [17, 19, 31]. Our previous work also followed this approach [9]. However, this type of approach is often associated with human overhead and bias. In addition, one may argue about the applicability of our sampling framework to other languages if we rely on an existing Japanese thesaurus for the similarity computation. In consideration of these problems, we adopted statistical models [4, 12, 29] for the purpose of this paper. Here, each noun n can be represented by a vector comprising statistical co-occurrence factors. This is expressed by equation (2), where $\vec{n}$ is the vector for the noun in question, and t_i is the co-occurrence statistics of n and each co-occurring verb.

$$\vec{n} = < t_1,\ t_2,\ \ldots,\ t_i,\ \ldots > \tag{2}$$

Co-occurrence data was extracted from the RWC text base RWC-DB-TEXT-95-1 [26]. This text base consists of 4 years worth of Mainichi Shimbun [27] newspaper articles, which have been automatically annotated with morphological tags. The total morpheme content is about 100 million. Instead of conducting full parsing on the text, several heuristics were used in order to obtain dependencies between complements (noun + case marker) and verbs in the form of tuples $<n, c, v>$. In regard to t_i, we used the notion of TF·IDF [8], in which t_i is calculated as in equation (3), where $f(<n, c, v>)$ is the frequency of the tuple $<n, c, v>$, $f(<c, v>)$ is the frequency of tuple $<c, v>$, and T is the total number of the tuples within the overall co-occurrence data.

$$t_i = f(<n, c, v>) \cdot \log \frac{T}{f(<c, v>)} \tag{3}$$

We then compute the similarity between nouns n_C and e by the cosine of the angle between the two vectors $\vec{n_C}$ and $\vec{e}$. This is realized by equation (4).

$$sim(n_C, e) = \frac{\vec{n_C} \cdot \vec{e}}{|\vec{n_C}||\vec{e}|} \tag{4}$$

3 Sampling algorithm

3.1 Overview

Let us look again at figure 1 in section 1. In this figure, "WSD outputs" refers to a corpus in which each sentence is assigned an expected interpretation of the verb during the WSD phase. In the training phase, the system stores supervised samples (with each interpretation simply checked or appropriately corrected by a human) in the database, to be used in a later WSD phase. In this section, we turn to the problem as to which examples should be selected as samples.

Lewis et al. proposed the notion of uncertainty sampling for the training of statistics-based text classifiers [18]. Their method selects those examples that the system classifies with minimum certainty, based on the assumption that there is no need for teaching the system the correct answer when it has answered with sufficiently high certainty. However, we should take into account the training effect a given example has on other remaining (unsupervised) examples. In other words, we would like to select samples such as to be able to correctly disambiguate as many examples as possible in the next iteration. If this is successfully done, the number of examples to be supervised will decrease. We consider maximization of

this effect by means of a training utility function aimed at ensuring that the example with the greatest training utility factor, is the most useful example at a given point in time. Intuitively speaking, the training utility of an example is greater when we can expect greater increase in the interpretation certainty of the remaining examples after training using that example.

Let S be a set of sentences, i.e. a given corpus, and D be the subset of supervised examples stored in the database. Further, let X be the set of unsupervised examples, realizing equation (5).

$$\mathbf{S} = \mathbf{D} \cup \mathbf{X} \tag{5}$$

The example sampling procedure can be illustrated as:

1. $WSD(\mathbf{D}, \mathbf{X})$
2. $e \leftarrow \arg\max_{x \in \mathbf{X}} TU(x)$
3. $\mathbf{D} \leftarrow \mathbf{D} \cup \{e\}, \quad \mathbf{X} \leftarrow \mathbf{X} \cap \overline{\{e\}}$
4. goto 1

where $WSD(\mathbf{D}, \mathbf{X})$ is the verb sense disambiguation process on input X using D as the database. In this disambiguation process, the system outputs the following for each input: (a) a set of verb sense candidates with interpretation scores, and (b) an interpretation certainty. These factors are used for the computation of $TU(x)$, newly introduced in our method. $TU(x)$ computes the training utility factor for an example x. The sampling algorithm gives preference to examples of maximum utility.

We will explain in the following sections how one can estimate $TU(x)$, based on the estimation of the interpretation certainty.

3.2 Interpretation certainty

Lewis et al. estimate certainty of an interpretation as the ratio between the probability of the most plausible text category, and the probability of any other text category, excluding the most probable one. Similarly, in our verb sense disambiguation system, we introduce the notion of interpretation certainty of examples based on the following preference conditions:

1. the highest interpretation score is greater,
2. the difference between the highest and second highest interpretation scores is greater.

The rationale for these conditions is given below. Consider figures 3 and 4, where each symbol denotes an example in a given corpus, with symbols x as unsupervised examples and symbols e as supervised examples. The curved lines delimit the semantic vicinities (extents) of the two verb senses 1 and 2, respectively[5]. The semantic similarity between two examples is graphically portrayed by the physical distance between the two symbols representing them. In figure 3, x's located inside a semantic vicinity are expected to be interpreted as being similar to the appropriate example e with high certainty, a fact which is in line with condition 1 above. However, in figure 4, the degree of certainty for the interpretation of any x which is located inside the intersection of the two semantic vicinities cannot be great. This occurs when the case fillers associated with two or more verb senses are not selective enough to allow for a clear cut delineation between them. This situation is explicitly rejected by condition 2.

Based on the above two conditions, we compute interpretation certainties using equation (6), where $C(x)$ is the interpretation certainty of an example x. $Score_1(x)$ and $Score_2(x)$ are the highest and second highest scores for x, respectively. λ, which ranges from 0 to 1, is a parametric constant used to control the degree to which each condition affects the computation of $C(x)$.

$$C(x) = \lambda \cdot Score_1(x) + (1 - \lambda) \cdot (Score_1(x) - Score_2(x)) \tag{6}$$

[5]Note that this method can easily be extended for a verb which has more than two senses. In section 4, we describe an experiment using multiply ambiguous verbs.

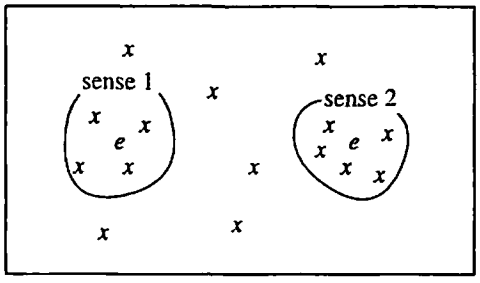

Figure 3: The case where the interpretation certainty of the enclosed x's is great

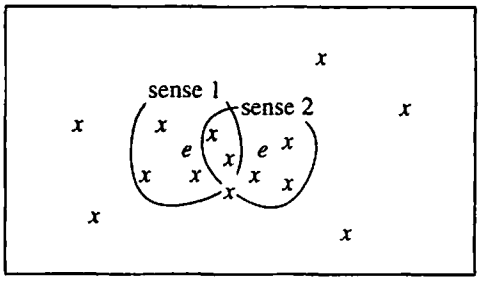

Figure 4: The case where the interpretation certainty of the x's contained in the intersection of senses 1 and 2 is small

3.3 Training utility

The training utility of an example a is greater than that of another example b when the total interpretation certainty of unsupervised examples increases more after training with example a than with example b. Let us consider figures 5 and 6, in which the x-axis mono-dimensionally denotes the semantic similarity between two unsupervised examples, and the y-axis denotes the interpretation certainty of each example. Let us compare the training utility of the examples a and b in figure 5. Note that in this figure, whichever example we use for training, the interpretation certainty for each unsupervised example (x) neighboring the chosen example increases based on its similarity to the supervised example. Since the increase in the interpretation certainty of a given x becomes smaller as the similarity to a or b diminishes, the training utility of the two examples can be represented by the shaded areas. It is obvious that the training utility of a is greater as it has more neighbors than b. On the other hand, in figure 6, b has more neighbors than a. However, since b is semantically similar to e, which is already contained in the database, the total *increase* in interpretation certainty of its neighbors, i.e. the training utility of b, is smaller than that of a.

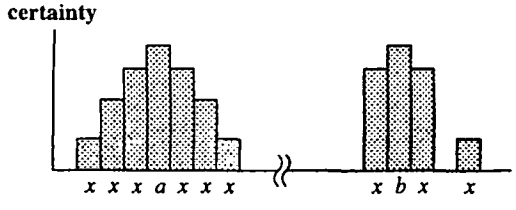

Figure 5: The case where the training utility of a is greater than that of b because a has more unsupervised neighbors

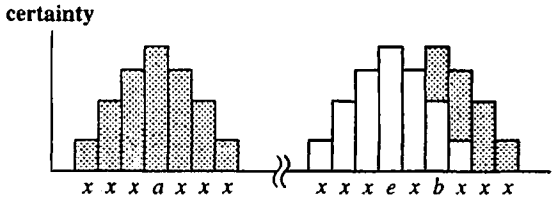

Figure 6: The case where the training utility of a is greater than that of b because b closely neighbors e, contained in the database

Let $\Delta C(x{=}s, y)$ be the difference in the interpretation certainty of $y \in \mathbf{X}$ after training with $x \in \mathbf{X}$, taken with the sense s. $TU(x{=}s)$, which is the training utility function for x taken with sense s, can be computed by way of equation (7)[6].

$$TU(x{=}s) = \sum_{y \in \mathbf{X}} \Delta C(x{=}s, y) \tag{7}$$

[6]It should be noted that in equation (7), we can replace $\mathbf{X}$ with a smaller subset of $\mathbf{X}$ which consists of neighbors of x: example y is a neighbor of x if they share the same supervised example e as their nearest neighbor. Since the system determines e during the WSD phase, identifying the neighbors of x does not require any extra computational overhead.

Since there is no guarantee that x will be supervised with any given sense s, it can be risky to rely solely on $TU(x=s)$ for the computation of $TU(x)$. We estimate $TU(x)$ by the expected value of x, calculating the average of each $TU(x=s)$, weighted by the probability that x takes sense s. This can be realized by equation (8), where $P(x=s)$ is the probability that x takes the sense s.

$$TU(x) = \sum_{s} P(x=s) \cdot TU(x=s) \tag{8}$$

Given the fact that (a) $P(x=s)$ is difficult to estimate in the current formulation, and (b) the cost of computation for each $TU(x=s)$ is not trivial, we temporarily approximate $TU(x)$ as in equation (9), where $\mathbf{K}$ is a set of the k-best verb sense(s) of x with respect to the interpretation score in the current state.

$$TU(x) \simeq \frac{1}{k} \sum_{s \in \mathbf{K}} TU(x=s) \tag{9}$$

3.4 Time complexity

The number of samples selected at each iteration should ideally be one, so as to avoid the supervision of similar examples. On the other hand, a small sampling size generates a considerable computation overhead for each iteration of the sampling procedure. This can be a critical problem for statistics-based approaches [4], as the reconstruction of statistic classifiers is expensive. However, fortunately, example-based systems [10, 17, 19, 31] do not require reconstruction, and examples simply have to be stored in the database.

Furthermore, in each disambiguation phase, our example-based system needs only compute the similarity between each newly stored example and its unsupervised neighbors, rather than between every example in the database and every unsupervised example. This reduces the time complexity of each iteration from $O(N^2)$ to $O(N)$, given that N is the total number of examples in a given corpus.

4 Comparative evaluation

In this experiment, we compared the following four example sampling methods by evaluating the relation between the number of samples and the performance of the system:

- a control (random), in which a certain proportion of a given corpus is randomly selected for training,
- uncertainty sampling (US), in which examples with minimum interpretation certainty are selected [18],
- committee-based sampling (CBS) [7],
- our method based on the notion of training utility (TU).

In committee-based sampling, the system selects samples based on the degree of disagreement between models randomly taken from a given set of supervised examples (these models are called "committees"). This is achieved by the iteratively repeating the steps given below, in which the number of committees is given as two without loss of generality[7]:

1. draw two committees randomly,
2. classify unsupervised example x according to each model, producing classifications C_1 and C_2,
3. if $C_1 \neq C_2$ (the committees disagree), select x for the training of the system.

We collected sentences containing eight frequently appearing verbs (as test/training data) from the EDR Japanese corpus [6] (originally produced from news articles). The input had to be both morphologically and syntactically analyzed prior to the sense disambiguation process. For this purpose, we

[7] Engelson et al. applied this framework to HMM training for part-of-speech tagging [7].

experimentally used the Japanese morph/syntax parser "QJP" [15], disallowing sentences which did not provide any complements of the target verb (in most cases, due to ellipsis or zero anaphora). The EDR corpus also provides sense information for each word, based on the EDR dictionary. However, since it was evident that sense classification criteria in the EDR dictionary were not necessarily clear even for human experts, we used verb senses described in the NTT semantic dictionary [25]. In this dictionary, each Japanese verb is subcategorized based on its English translations, and thus sense classification can be achieved more impartially. We then removed sentences in which the target verb comprised an idiomatic phrase[8]. As a result, the total number of sentences was 3185, and the average number of sense candidates for each verb was 5.4.

We conducted 4-fold cross validation, that is, we divided the training/test data into four equal parts, and conducted four trials in which a different part was used as test data each time, and the rest as training data, from which each sampling method selected samples. We evaluated each system's performance according to its accuracy, that is the ratio of the number of correct outputs, compared to the total number of inputs contained in the test data. For the purpose of this experiment, we set $\lambda = 0.5$ for equation (6) so that the both preference conditions (see section 3.2) were equally reflected in the computation, and $k = 1$ for equation (9)[9]. To initialize the system for any sampling methods, we randomly selected one example (seed) for each verb sense. Figure 7 shows the relation between the size of the training data and the precision of the system. In figure 7, zero on the x-axis represents the system using only seeds. Looking at figure 7 one can see that our method generally reduced the size of the training data (that is, overhead for supervision and overhead for searching the database) required to achieve any given accuracy compared with other three methods. For example, to achieve an accuracy of 80%, the size of the training data required for our method was roughly one-forth of that for random sampling. This tendency was also observed in our previous experiments [9]. Surprisingly, the result of random sampling was almost equivalent to that of committee-based sampling, and uncertainty sampling performed worst.

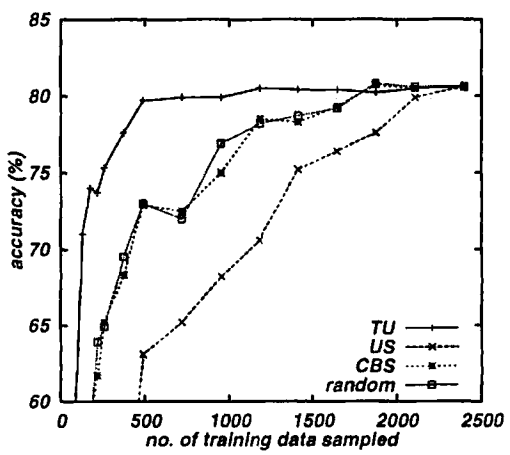

Figure 7: The relation between the number of training data sampled and accuracy of the system

5 Discussion

First, let us discuss the theoretical difference between uncertainty sampling and our method. Considering figures 5 and 6 again, one can see that the concept of training utility is supported by the following properties:

[8]For practical purposes, idiomatic expressions should be separately described in the database so that the system can control their overgeneralization [31].

[9]Based on a preliminary experiment, increasing the value of k either did not improve the performance over that for $k = 1$.

1. an example which neighbors more unsupervised examples is more informative (figure 5),

2. an example less similar to one already existing in the database is more informative (figure 6).

Uncertainty sampling directly addresses property 2, but ignores property 1. It differs from our method more crucially when more unsupervised examples remain, because these unsupervised examples have a greater influence on the computation of training utility. This assumption can also be seen in the comparative experiments in section 4, in which our method outperformed uncertainty sampling to the highest degree in early stages.

Second, figure 8 shows a typical disparity evident between committee-based sampling and our sampling method. The basic notation in this figure is the same as in figures 3 and 4, and both x and y denote unsupervised examples, or more formally $\mathbf{D} = \{e_1, e_2\}$, and $\mathbf{X} = \{x, y\}$. Assume a pair of committees $\{e_1\}$ and $\{e_2\}$ have been selected from the database $\mathbf{D}$. In this case, the committees disagree as to the interpretations of both x and y, and consequently, both examples can potentially be selected as a sample for the next iteration. In fact, committee-based sampling tends to require a number of similar examples (similar to e_1 and y) in the database, otherwise committees taken from the database will never agree. This feature provides a salient contrast to our method for which a similar example is less informative, and x is therefore preferred to y as a sample. This contrast can also correlate to the fact that committee-based sampling is currently applied to statistics-based language models (HMM classifiers), in other words, statistical models generally require that the distribution of the training data reflects that of the overall text. Through this argument, one can assume that committee-based sampling is better suited to statistics-based systems, while our method is more suitable for example-based systems.

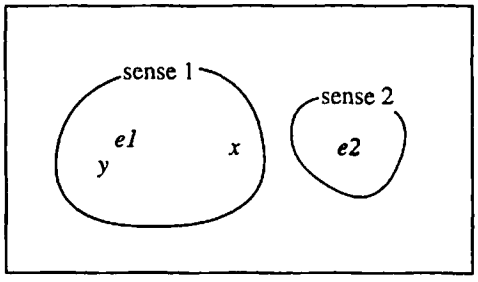

Figure 8: A case where either x or y can be selected in committee-based sampling

6 Conclusion

In this paper, we suggested the possibility of applying word sense disambiguation to information extraction. A shortcoming of recent corpus-based word sense disambiguation techniques has been that they tend to require a considerable overhead for supervision in constructing a large-sized database, additionally resulting in a computational overhead to search the database. To overcome these problems, our method selectively samples a smaller-sized subset from a given example set.

We reported on the performance of our proposed sampling method by way of experiments, in which we compared our method with random sampling, uncertainty sampling [18] and committee-based sampling [7]. As far as the corpus we used were concerned, our method reduced both the overhead for supervision and the overhead for searching the database to a larger degree than any of the above methods, without degrading the performance of verb sense disambiguation.

Future work will include empirical evaluation of the application of our word sense disambiguation technique to information extraction systems.

Acknowledgments

The authors would like to thank Dr. Manabu Okumura (JAIST, Japan), Dr. Michael Zock (LIMSI, France), Dr. Dan Tufis (Romanian Academy, Romania), and anonymous reviewers for their comments on an earlier version of this paper, and Mr. Masayuki Kameda (RICOH Co., Ltd., Japan) for his support with the QJP parser, and Mr. Naoyuki Sakurai (TITECH, Japan) for aiding with experiments, and NTT staff (NTT, Japan) for their support with the NTT semantic dictionary.

References

[1] David W. Aha, Dennis Kibler, and Marc K. Albert. Instance-based learning algorithms. *Machine Learning*, Vol. 6, No. 1, pp. 37–66, 1991.

[2] Peter F. Brown, Stephen A. Della Pietra, and Vincent J. Della Pietra. Word-sense disambiguation using statistical methods. In *Proceedings of ACL*, pp. 264–270, 1991.

[3] Robert L. Chapman. *Roget's International Thesaurus (Fourth Edition)*. Harper and Row, 1984.

[4] Eugene Charniak. *Statistical Language Learning*. MIT Press, 1993.

[5] David Cohn, Les Atlas, and Richard Ladner. Improving generalization with active learning. *Machine Learning*, Vol. 15, No. 2, pp. 201–221, 1994.

[6] EDR. *EDR Electronic Dictionary Technical Guide*, 1995. (In Japanese).

[7] Sean P. Engelson and Ido Dagan. Minimizing manual annotation cost in supervised training from corpora. In *Proceedings of ACL*, pp. 319–326, 1996.

[8] William B. Frankes and Ricardo Baeza-Yates. *Information Retrieval: Data Structure & Algorithms*. PTR Prentice-Hall, 1992.

[9] Atsushi Fujii, Kentaro Inui, Takenobu Tokunaga, and Hozumi Tanaka. Selective sampling of effective example sentence sets for word sense disambiguation. In *Proceedings of the Fourth Workshop on Very Large Corpora*, pp. 56–69, 1996. `http://xxx.lanl.gov/ps/cmp-lg/9702010`.

[10] Atsushi Fujii, Kentaro Inui, Takenobu Tokunaga, and Hozumi Tanaka. To what extent does case contribute to verb sense disambiguation? In *Proceedings of COLING*, pp. 59–64, 1996.

[11] Fumiyo Fukumoto and Jun'ichi Tsujii. Automatic recognition of verbal polysemy. In *Proceedings of COLING*, pp. 764–768, 1994.

[12] Donald Hindle. Noun classification from predicate-argument structures. In *Proceedings of ACL*, pp. 268–275, 1990.

[13] Graeme Hirst. *Semantic Interpretation and the Resolution of Ambiguity*. Cambridge University Press, 1987.

[14] Hiroyuki Kaji, Yuuko Kida, and Yasutsugu Morimoto. Learning translation templates from bilingual text. In *Proceedings of COLING*, pp. 672–678, 1992.

[15] Masayuki Kameda. A portable & quick Japanese parser : QJP. In *Proceedings of COLING*, pp. 616–621, 1996.

[16] Robert Krovets and W. Bruce Croft. Lexical ambiguity and information retrieval. *ACM Transactions on Information Systems*, Vol. 10, No. 2, pp. 115–141, 1992.

[17] Sadao Kurohashi and Makoto Nagao. A method of case structure analysis for Japanese sentences based on examples in case frame dictionary. *IEICE TRANSACTIONS on Information and Systems*, Vol. E77-D, No. 2, pp. 227–239, 1994.

[18] David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers. In *Proceedings of ACM SIGIR*, pp. 3–12, 1994.

[19] Xiaobin Li, Stan Szpakowicz, and Stan Matwin. A WordNet-based algorithm for word sense disambiguation. In *Proceedings of IJCAI*, pp. 1368–1374, 1995.

[20] Steven L. Lytinen. Dynamically combining syntax and semantics in natural language processing. In *Proceedings of AAAI*, pp. 574–578, 1986.

[21] George A. Miller, et al. Five papers on WordNet. Technical report, Cognitive Science Laboratory, Princeton University, 1993.

[22] Katashi Nagao. A preferential constraint satisfaction technique for natural language analysis. *IEICE TRANSACTIONS on Information and Systems*, Vol. E77-D, No. 2, pp. 161–170, 1994.

[23] National Language Research Institute. *Bunruigoihyo*, revised and enlarged edition, 1996. (In Japanese).

[24] Hiroshi Nomiyama. Machine translation by case generalization. *Information Processing Society of Japan*, Vol. 34, No. 5, pp. 905–912, 1993. (In Japanese).

[25] NTT. *NTT Semantic Dictionary*.

[26] Real World Computing Partnership. RWC text database. `http://www.rwcp.or.jp/wswg.html`, 1995.

[27] Mainichi Shimbun. Mainichi shimbun CD-ROM '91-'94, 1991-1994.

[28] Barry Smyth and Mark T. Keane. Remembering to forget: A competence-preserving case deletion policy for case-based reasoning systems. In *Proceedings of IJCAI*, pp. 377–382, 1995.

[29] Takenobu Tokunaga, Makoto Iwayama, and Hozumi Tanaka. Automatic thesaurus construction based on grammatical relations. In *Proceedings of IJCAI*, pp. 1308–1313, 1995.

[30] Naohiko Uramoto. A best-match algorithm for broad-coverage example-based disambiguation. In *Proceedings of COLING*, pp. 717–721, 1994.

[31] Naohiko Uramoto. Example-based word-sense disambiguation. *IEICE TRANSACTIONS on Information and Systems*, Vol. E77-D, No. 2, pp. 240–246, 1994.

[32] Takehito Utsuro. Sense classification of verbal polysemy based-on bilingual class/class association. In *Proceedings of COLING*, pp. 968–973, 1996.

[33] Ellen M. Voorhees. Using WordNet to disambiguate word senses for text retrieval. In *Proceedings of ACM SIGIR*, pp. 171–180, 1993.

[34] David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of ACL*, pp. 189–196, 1995.