The Use of Word Sense Disambiguation in an Information Extraction System

Joyce Yue Chai
30 Saw Mill River Rd.
IBM T. J. Watson Research Center
Hawthorne, NY 10532
jchai@us.ibm.com

Alan W. Biermann
Computer Science Department
Duke University
Durham, NC 27708
awb@cs.duke.edu

Abstract

This paper describes a rule-based methodology for word sense disambiguation and an application of the methodology to information extraction using rules generalized with the help of the WordNet system. The methodology creates word sense disambiguation rules based on user trained examples working in the domain of interest. It achieves accuracy rates comparable to the best competing methods and can be easily integrated into higher level applications.

Introduction

Most information extraction (IE) systems have used hand-crafted semantic resources for each application domain, or have employed techniques for automatically or semi-automatically constructing lexicons of annotated texts in the domain (Riloff & Lehnert 1993) (Riloff 1996) (Krupka 1995). Few examples apply general lexical semantic resources. NYU's MUC-4 system (Grishman, Macleod, & Sterling 1992) made some attempt at using WordNet for semantic classification. However, they ran into the problem of automated sense disambiguation because the WordNet hierarchy is sense dependent. Are the generic lexical semantic databases useful at all for information extraction purposes? Can we avoid the process of hand-crafting domain specific knowledge? If so, how can we effectively use the generic lexical semantics?

In order to apply generic resources effectively, sense selection among the polysemies becomes important. There is a great amount of work concerning Word Sense Disambiguation (WSD), especially various algorithms that can improve the accuracy of WSD on a predefined set of words in pre-annotated corpora. Some recent work on WSD includes the use of the knowledge contained in a machine-readable dictionary (Luk 1995)(Wilks *et al.* 1990), supervised learning from tagged sentences (Bruce & Wiebe 1994)(Miller 1990)(Ng & Lee 1996)(Yarowsky 1992)(Yarowsky 1994), unsupervised learning from raw corpora (Yarowsky 1995), and hybrid methods that combine several knowledge sources, collocation and others (Bruce & Wiebe 1994) (Ng & Lee 1996). Despite the strides made in developing robust WSD algorithms, its use as a technology in higher level applications has been limited. Some work has shown WSD could help with information retrieval and machine translation. However, its role in information extraction has not been investigated. Furthermore, even with the best algorithm, the question arises as to whether that algorithm is applicable to real NLP systems, since most algorithms heavily depend on large corpora of annotated text which might not be available.

In this paper, we will describe an information extraction system that applies a generic lexical semantic resource (WordNet) and achieves competitive results. In particular, we will give a detailed description of a WSD algorithm that is integrated into the system and makes the application of generic lexical semantics possible. First, we will give a brief introduction to the system. Then we will present a rule-based WSD algorithm and its performance on a common data set. Finally, we will describe the integration of this WSD algorithm into the information extraction system and its performance.

TIMES System

The *T*rainable *I*nfor*M*ation *E*xtraction *S*ystem (TIMES) is shown in diagram form in Figure 1 where the main processors handle Tokenization, Lexical Processing, Partial Parsing, and Rule Learning and Generalization (Chai 1998). The Tokenizer segments the input text into words and sentences. The Lexical Processing stage tags words with syntactic information from CELEX database[1] and semantic information from WordNet (Miller 1990). In particular, a Semantic Type Identifier is used to identify special semantic types such as email and web addresses, file and directory names, dates, times, and dollar amounts, telephone numbers, zip codes, cities, states, countries, names of companies, and many others. The Partial Parser uses a fi-

[1]CELEX was developed by several universities and institutions in the Netherlands, and is distributed by the Linguistic Data Consortium.

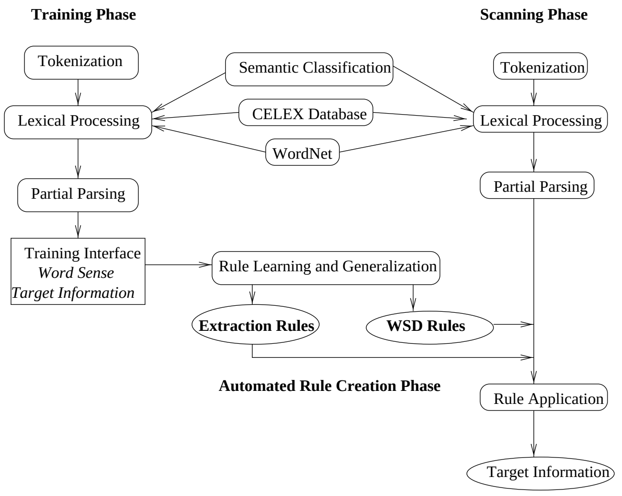

Figure 1: System Overview

nite state model to discover noun phrases (NG), verb phrases (VG), and prepositions (PG). Verb phrases are further categorized as Verb Active (VG), Verb Passive (VG_ED), Verb Be (VG_BE), Verb Gerund (VG_ING), and Verb Infinitive (VG_TO). The headword of a phrase is the base form of the last word in the phrase. The Rule Learning and Generalization module constructs extraction rules based on information given by the user through the training interface and on the preprocessing of the earlier stages.

The TIMES system has three modes of operation which implement, respectively, training, rule creation, and text scanning. The first mode enables the user to move through a document and indicate desired information items in the text. All significant actions by the user are gathered by the system and used for generating rules. One of the necessary inputs by the user is a designation of the WordNet word senses for the purposes of training the system. However, the system requires only minimum linguistic and domain knowledge so that casual users should be able to use it. The rule creation phase generates rules, as described below, for information extraction and for word sense disambiguation. Finally, the scanning phase applies the created rules to any body of free text for the purposes of information extraction.

Training and Generalization for Extraction Rules

TIMES provides users a convenient interface for training. The interface shows a stack of phrases segmented from the sentence by the Partial Parser. Associated with each phrase, the syntactic category, semantic type, headword, and meanings (i.e., senses) for the headword are also given. The user is required to first define the information of interest (i.e., the target information), and then identify some phrases as one type of target information. Furthermore, the user needs to specify the correct sense for the headword. (The training interface provides sense definitions and examples so that the user would know which sense to choose.) By default, TIMES assigns sense one, the most frequently used sense in WordNet to the headword if no specification from the user is given. In addition, if the user is experienced, he/she can assist the system to learn by specifying important phrases in the sentence that are crucial to the rule generation. However, this specification is optional. If the user does not identify the important phrases, TIMES will regard every phrase as an important phrase.

Extraction rules are generated based on the important phrases and user specified target information. For each important phrase in the training sentence, if its headword exists in WordNet, then together with its syntactic category and sense number, it can uniquely determine a concept (i.e., a synonym list) in WordNet. By following the hypernym path in WordNet, the concept becomes more and more general (Chai & Biermann 1997).

Extraction rules are pattern-action rules. The pattern, defined by the left hand side (LHS) of a rule, is a conjunction (expressed by $\wedge$) of *subsumption functions* $S(X, \alpha, target(\alpha))$. X is instantiated by a new phrase when the rule is applied; α is the concept corresponding to the headword of an important phrase in the training sentence; and $target(\alpha)$ is the type of target information identified for α. The action in the right hand side (RHS) of a rule, $FS(X, target(\alpha))$ fills a template slot, assigning the type of target information $target(\alpha)$ to the phrase X.

The subsumption function $S(X, \alpha, target(\alpha))$ looks for subsumption of concepts. It returns true if the headword of X is subsumed to the concept α. If all subsumption functions on the LHS return true, then the RHS action will take place to extract X as a type $target(\alpha)$.

The system applies a two-dimensional generalization model to learn the number of subsumption functions to be included in the LHS, the order of subsumption functions, and specification/generalization of α (from WordNet hierarchy) in the subsumption function (Chai, Biermann, & Guinn 1999). For example, the user is interested in finding out which company has what position open. Suppose the training sentence is "IBM is looking for software engineers." Through the interface, the user identifies "IBM" as the target information *COMPANY* and "software engineers" as the target information *POSITION*. Based on this input, the system can generate a set of rules. Some sample rules are shown in Figure 2. Since two types of target information are specified, two rules are necessary to capture both *COMPANY* and *POSITION* as shown in R_1 and R_2. (By our convention, each rule only captures one type of target information.) In Figure 2, R_3 is more general than R_1 since it has fewer constraints on the LHS. R_4 is more general than R_1 since the subsumed concepts in R_4 are more general than those in R_1. For each type of target

R_1: $S(X_1, \{company\}, COMPANY) \wedge S(X_2, \{look\_for\}, none) \wedge S(X_3, \{engineer\}, POSITION)$
$\longrightarrow FS(X_1, COMPANY)$
R_2: $S(X_1, \{company\}, COMPANY) \wedge S(X_2, \{look\_for\}, none) \wedge S(X_3, \{engineer\}, POSITION)$
$\longrightarrow FS(X_3, POSITION)$
R_3: $S(X_1, \{company\}, COMPANY) \wedge S(X_2, \{look\_for\}, none) \longrightarrow FS(X_1, COMPANY)$
R_4: $S(X_1, \{group, ...\}, COMPANY) \wedge S(X_2, \{look\_for, \}, none) \wedge S(X_3, \{professional, ...\}, POSITION)$
$\longrightarrow FS(X_1, COMPANY)$

Figure 2: Examples of Extraction Rules

information, the system will automatically determine the generalization/specification for rules.

The Semantic Tagging Approach

Successful semantic generalization by use of WordNet conceptual hierarchy depends on the correct word senses. Semantic tagging (and the associated word sense disambiguation) has been studied in great detail. However, its use in higher level applications is limited. We have designed a WSD algorithm which can be integrated into information extraction systems, particularly IE systems which use generic lexical semantic resources. In this section, we will describe the algorithm and report on its performance on a common data set.

The Learning Model for WSD Rules

Our rule learning model follows the supervised learning paradigm. The model automatically generates useful rules from the tagged examples for WSD tasks. A *target word* is defined as the word that needs to be sense identified. The WSD rules will be generated based on the context of the target word.

Context After being processed by the Partial Parser, each pre-tagged sentence is divided into a sequence of phrases. Each phrase corresponds to a segment group $g = (phrase, head, syn\_type, sem\_type, sense)$, where *phrase* is the phrase itself; *head* is the headword of the phrase; *syn_type* is the syntactic category of the phrase; *sem_type* is the semantic type identified by the Semantic Type Identifier; *sense* is the pre-tagged sense number. If a phrase is identified as a special semantic type, then its headword is the name for that special type. Since we are only interested in nouns and verbs, the target word is always the headword of a phrase.

Within a sentence, the context of a target word is created based on the surrounding phrases. More precisely, if g_t contains a target word (where t is the position/index of the phrase in the sentence, $1 \leq t \leq n$), then the context of g_t, with contextual range d, is $(g_i, ..., g_{t-1}, g_t, g_{t+1}, ..., g_j)$, where $i = \max(t-d, 1)$, $j = \min(t+d, n)$, and each g_i corresponds to one phrase in the sentence.

For example, suppose the original sentence is: "Cray Research will retain a 10 percent interest in the new company, which will be based in Colorado Springs, CO." The sense for "interest" is pre-tagged as "5". After applying the Partial Parser and the Semantic Type Identifier, the system will mark the sentence as: "[Cray Research/NG/Company_type] [will retain/VG] [a 10 percent/NG] [interest_5/NG] [in/PG] [the new company/NG] which [will be based/VG_ED] [in/PG] [Colorado Springs/NG/City_type] [CO./NG/State_type]." The string "A 10 percent interest" is a noun phrase parsed by the Partial Parser. However, due to the assumption that noun modifiers can be a good indicator for the sense of the target word, in our approach, the target word (if it is a noun) is separated from its noun modifiers.

Creating WSD Rules In general, rules consist of a left hand side (LHS) which defines the applicability conditions and a right hand side (RHS) which specifies the action to select senses. The LHS of a rule is a conjunction (expressed by $\wedge$) of *Match Functions*. Each Match Function corresponds to one phrase in the context of the target word. Given $g = (phrase, head, syn\_type, sem\_type, sense)$ as previously defined, the system creates a Match Function $Match(X, head, syn\_type, sense)$, where X is the variable field to be filled by a new phrase. $Match(X, head, syn\_type, sense)$ returns true if headword of X is the same as *head* and the syntactic category of X is the same as *syn_type*. The RHS $T(X, sense)$ is the sense tagging function which identifies the sense of the headword of X as *sense*. The number of Match Functions on the LHS indicates specification/generalization of the rule. More numbers imply more constraints, and therefore, a more specific rule.

For each context of the target word, a rule with n Match Functions can be generated by simply selecting n phrases (including the one containing a target word) from the context, and creating corresponding Match Functions to make up the LHS. The RHS of the rule is generated according to the correct sense for the target word in that particular context. Some examples of WSD rules are shown in Figure 3.

The first rule in Figure 3 is very precise; however, the chance of obtaining a match in the unseen data is small. On the other hand, the fourth rule is too general. Applying this rule simply tags all words with one sense, and the highest performance would be no better than that attained by identifying all words with the most frequently used senses. Based on this scenario, we would like to generate rules with the optimum num-

1. $Match(X_1, retain, VG, 1) \wedge Match(X_2, percent, NG, 1) \wedge Match(X_3, interest, NG, 5)$
   $\wedge Match(X_4, in, PG, 1) \wedge Match(X_5, company, NG, 1) \longrightarrow T(X_3, 5)$
2. $Match(X_1, retain, VG, 1) \wedge Match(X_2, percent, NG, 1) \wedge Match(X_3, interest, NG, 5) \longrightarrow T(X_3, 5)$
3. $Match(Match(X_1, percent, NG, 1) \wedge Match(X_2, interest, NG, 5) \longrightarrow T(X_2, 5)$
4. $Match(X_1, interest, NG, 5) \longrightarrow T(X_1, 5)$

Figure 3: Examples of WSD Rules

ber of Match Functions. This process is carried out by checking the *Precision Rate* for each rule.

Suppose a rule r_i is applied to the training data, precisely, to the context of a target word. If all the entities on the LHS are satisfied, then a particular sense will be assigned to the target word. By comparing the senses identified with the original tagged senses, the Precision Rate $P(r_i)$ is derived:

$$P(r_i) = \frac{number\ of\ correct\ senses\ identified\ by\ r_i}{number\ of\ senses\ identified\ by\ r_i}$$

A threshold θ is predefined to select useful rules. If $P(r_i) > \theta$, r_i is considered a useful rule and will be applied for WSD on unseen data.

We applied a greedy covering algorithm to generate WSD rules:

1. Predefine N as the maximum number of Match Functions allowed in a rule, d as the contextual range, and θ as the threshold.

2. For each target word, based on its context with contextual range d, create all possible rules with one Match Function, two Match Functions, ..., and up to N Match Functions.

3. For each rule r_i, apply it to the training data and compute $P(r_i)$. If $P(r_i) > \theta$, put r_i in the rule base.

4. Sort the rules to ensure no repetition of rules.

Applying the Rules When useful rules are applied to new sentences, two types of match routines are used. The full match routine guarantees that only when each Match Function in the LHS of a rule returns true, does the RHS take place. The partial match routine allows a rule with N Match Functions to be activated when only $N - 1$ Match Functions are satisfied. The application of rules takes place as follows (assuming new sentences have been properly pre-processed):

- For each context of the target word, let $n = N$. Start applying rules with n Match Functions. In doing so, it first applies rules with the Precision Rate between 0.9 to 1.0. If there are matches, it selects the sense which is identified by the most of rules and proceeds to the next context. Otherwise, it applies the rules with the Precision Rate between 0.8 to 0.9. The procedure continues. If there are matches, then it assigns the sense and proceeds to the next context; otherwise it decrements the Precision Rate by 0.1 until it reaches the threshold.
- If there is no match, it applies rules with $n = n - 1$ Match Functions with the decremental Precision Rate. If there are matches, it then assigns the sense which is identified by the most of rules and proceeds to the next context; otherwise it applies rules with fewer Match Functions until n becomes 0.
- If there is no match, it operates the partial match to the rules with N entities and selects the sense which is identified by the most of those partial matches.
- If there is no match, the most frequently used sense will be assigned.

The approach for applying rules will first achieve the highest precision for a small number of identifications. Then by applying rules with decremented Precision Rate and fewer Match Functions, more identifications will take place while maintaining the overall precision.

A Test on a Common Data Set

We chose a common data set (Bruce & Wiebe 1994) involving the noun "interest" to test the performance of our WSD algorithm. This data set was extracted from Penn Treebank (Marcus, Santorini, & Marcinkiewicz 1993) and made available by Bruce and Wiebe (Bruce & Wiebe 1994). It consisted of 2369 sentences and each sentence had an occurrence of the noun "interest" tagged with a correct sense. The sense definitions used were from the Longman Dictionary of Contemporary English (LDOCE). The six senses of the noun "interest" are: 1) readiness to give attention (15% in the dataset); 2) quality of causing attention to be given (about 1%); 3) activity, subject, etc. which one gives time and attention to (3%); 4) advantage, advancement, or favor (8%); 5) a share in a company, business, etc. (21%); 6) money paid for the use of money (53%). Senses 2 and 3 are dramatically underrepresented in the data set. Before using the WSD algorithm to learn the rules, all sentences are preprocessed by the Partial Parser and the Semantic Type Identifier. In the experiment, each trial included a randomly selected 1769 sentences as training data and 600 sentences as testing data. This is the same experimental strategy as described in (Ng & Lee 1996).

We ran 100 trials. In each trial, rules were created based on contextual range 2, and threshold 0.6. The average precision for identifying all six senses was 88.2%, with a standard deviation 1.0%; the average precision for identifying sense 1, 4, 5, 6 was 89.7%, with a standard deviation of 1.0%. The average precision and the standard deviation on each sense tagging is shown in Table 1.

sense	average precision	standard deviation
1	67.8%	4.1%
2	0	0
3	52.7%	10.2%
4	70.9%	7.6%
5	88.7%	2.6%
6	99.0%	0.6%
Overall	88.2%	1.0%
1,4,5,6	89.7%	1.0%

Table 1: Performance for WSD on Individual Sense

Many research groups have investigated WSD of the word "interest." In identifying four senses of the noun "interest." Black achieved 72% (Black 1988) and Yarowsky achieved 72% (Yarowsky 1992) . However, their work was not based on the same data set. Bruce and Weibe made this common data set available (Bruce & Wiebe 1994). They developed a decomposable probabilistic model (Bruce & Wiebe 1994), which used parts of speech and morphological forms for the surrounding words. Their model achieved 78% precision on identifying all six senses, and 79% on identifying senses 1, 4 5, and 6. Based on the same data set, Ng and Lee integrated multiple knowledge sources and used exemplar learning to achieve WSD (Ng & Lee 1996). In addition to parts of speech and morphological forms, they also took local collocation (common expressions) and verb-object syntactic relationships as features. Their LEXAS system achieved 87.4% in precision on all six senses of "interest"and 89% on senses 1, 4, 5, 6. Our rule learning model achieves the comparable performance (88.2% for all senses, 89.7% for senses 1, 4, 5, 6) in this common data set. It applies limited knowledge sources (syntactic category and preliminary semantic type classification). Furthermore, the model is ready to disambiguate senses for any target word.

Experimental Results

We have conducted experiments to test the applicability of our rule-based WSD algorithm in TIMES. The working domain is the *triangle.job* newsgroup, where job advertisements are posted. The types of the target information are defined as the following: *COMPANY* (the name of the company which has job openings), *POSITION* (the name of the available position), *SALARY* (the salary, stipend, compensation information), *LOCATION* (the state/city where the job is located), *EXPERIENCE* (years of experience), *CONTACT* (the phone number or email address for contact), *SKILLS* (the specific skills required, such as programming languages, operating systems, etc), *BENEFITS* (the benefits provided by the company, such as health, dental insurance, etc). The training set consisted of 24 articles and the testing set had 40 articles. As described earlier, in the training phase, the user is required to annotate the target information and tag the correct senses to the words which are not used as sense one. The average training time for each article was about three minutes. Based on each target word (the word which sense is not used as sense one), the system generated a set of WSD rules. In the scanning phase, WSD rules are applied to assign senses to target words. Furthermore, from the training examples, a set of extraction rules were generated to be applied in the scanning phase.

Based on the threshold 0.8 and the contextual range 2, the system generated a set of rules for the WSD task. Among those, three rules had three Match Functions, 21 rules had two Match Functions, and 10 rules had one Match Function. However, all rules with one Match Function could cover the rest of the rules since the Precision Rate for them is already very high. This observation suggested that, in a specific domain, senses of words tend to remain the same throughout the domain. For example, "position" has fifteen senses in WordNet, but in the *triangle.job* domain, every time it appears, it's always used as sense six which means a job in an organization.

The end performance of the system on extracting target information with and without WSD is shown in Table 2, The use of the WSD algorithm pushes up the overall performance in terms of F-measure by 7.5%. It is extremely helpful in enhancing recall (about 10%). This indicates that instead of building a domain specific knowledge base for information extraction, WSD can enable the use of an off-the-shelf lexical semantic resource. Our rule based WSD algorithm can be easily incorporated into such a system.

	with WSD	without WSD
precision	71.0%	67.2%
recall	67.5%	57.1%
F-measure.	69.2%	61.7%

Table 2: End Performance with and without WSD

Discussion

Since the first version of the system (Bagga, Chai, & Biermann 1997), TIMES has been upgraded in many ways. First, in the old system, the user created semantic transitions by specifying nodes and relations. The new version of the system replaces that training approach by asking users to indicate the target information and allowing the system to build rules automatically. Second, the old system only applied semantic generalization with various degrees and it didn't provide an automated mechanism to control the degree of generalization based on the training data. The new system automatically generates rules based on both syntactic generalization and semantic generalization. Furthermore the new approach determines the optimum amount of generalization for both directions. Finally, the new system provides a new framework to learn WSD rules in IE context.

By allowing the user to select the correct senses, the system can automatically generate WSD rules. Based on the assumption that most senses are used as sense one in WordNet and senses tend to remain the same for a specific domain, the sense training process is easier than the creation of a specific domain knowledge base. This process does not require the expertise in a particular domain and allows any casual user to accomplish it given a set of sense descriptions (glosses). We feel that, if possible, using generic resources is more efficient than hand-crafting domain specific knowledge with the respect to easy customization. Furthermore, in order to make the generic lexical semantic resources useful, word sense disambiguation is necessary, and moreover, an easily adaptable WSD approach is important.

In contrast with many statistically based WSD algorithms, our rule-based approach incorporates syntactic features and basic semantic knowledge. This approach has achieved comparable results on a common data set. Furthermore, the model is applicable to any target word and can be easily integrated into any system (not just information extraction systems) where large annotated corpora are not available. Finally, the rules learned can reflect the domain characteristics and allow easy interpretation.

Conclusion

In this paper, we have presented a WSD method in an information extraction system that uses WordNet for automated rule generalization. Furthermore, it demonstrates that, to successfully make use of the generic resources, WSD is very important. This calls for an adaptable WSD approach, and our rule based WSD algorithm meets this need.

Acknowledgments

We would like to thank Amit Bagga for developing the Tokenizer and the Semantic Type Identifier. We would also like to thank Jerry Hobbs for providing us with the finite state rules for the Partial Parser. This work was supported in part by an IBM Fellowship.

References

Bagga, A.; Chai, J.; and Biermann, A. 1997. The role of WordNet in the creation of a trainable message understanding system. *Proceedings of Ninth Conference on Innovative Applications of Artificial Intelligence (IAAI-97).*

Black, E. 1988. An experiment in computational discrimination of english word senses. *IBM Journal of Research and Development* 32(2).

Bruce, R., and Wiebe, J. 1994. Word sense disambiguation using decomposable models. *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics.*

Chai, J., and Biermann, A. 1997. Corpus based statistical generalization tree in rule optimization. *Proceedings of Fifth Workshop on Very Large Corpora (WVLC-5).*

Chai, J.; Biermann, A.; and Guinn, C. 1999. Two dimensional generalization in information extraction. *Proceedings of Sixteenth National Conference on Artificial Intelligence.*

Chai, J. 1998. *Learning and Generalization in the Creation of Information Extraction Systems.* Ph.D. Dissertation, Department of Computer Science, Duke University.

Grishman, R.; Macleod, C.; and Sterling, J. 1992. New York University Proteus system: MUC-4 test results and analysis. *Proceedings of the Fourth Message Understanding Conference.*

Krupka, G. 1995. Description of the SRA system as used for MUC-6. *Proceedings of the Sixth Message Understanding Conference.*

Luk, A. K. 1995. Statistical sense disambiguation with relatively small corpora using dictionary definitions. *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics.*

Marcus, M.; Santorini, B.; and Marcinkiewicz, M. 1993. Building a large annotated corpus of english: the Penn Treebank. *Computational Linguistics* 19(3).

Miller, G. 1990. WordNet: An on-line lexical database. *International Journal of Lexicography.*

Ng, H., and Lee, H. 1996. Integrating multiple knowledge sources to disambiguate word sense: An exemplar-based approach. *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics.*

Riloff, E., and Lehnert, W. 1993. Automated dictionary construction for information extraction from text. *Proceedings of Ninth IEEE Conference on Artificial Intelligence for Applications.*

Riloff, E. 1996. An empirical study of automated dictionary construction for information extraction in three domains. *AI Journal* 85.

Wilks, Y.; Fass, D.; Guo, C.; McDonald, J.; Plate, T.; and Slator, B. M. 1990. Providing machine tractable dictionary tools. *Machine Translation* 5(2).

Yarowsky, D. 1992. Word-sense disambiguation using statistical models of Roget's categories trained on large corpora. *Proceedings of the Fifteenth International Conference on Computational Linguistics.*

Yarowsky, D. 1994. Decision lists for lexical ambiguity resolution: Application to accent restoration in Spanish and French. *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics.*

Yarowsky, D. 1995. Unsupervised word sense disambiguation rivaling supervised methods. *Proceedings of the 33rd Association of Computational Linguistics.*