Evaluation of a Generic Lexical Semantic Resource in Information Extraction

Joyce Yue Chai

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

Abstract

We have created an information extraction system that allows users to train the system on a domain of interest. The system helps to maximize the effect of user training by applying WordNet to rule generation and validation. The results show that, with careful control, WordNet is helpful in generating useful rules to cover more instances and hence improve the overall performance. This is particularly true when the training set is small, where F-measure is increased from 65% to 72%. However, the impact of WordNet diminishes as the size of training data increases. This paper describes our experience in applying WordNet to this system and gives an evaluation of such an effort.

1. Introduction

Lexical information is important for natural language processing. It provides grammatical, syntactic, and semantic information about individual words or word strings (Guthrie et al., 1996). For some applications, lexical information can be acquired from the existing repositories of lexical knowledge, i.e., generic lexical resources. Those resources are generally in the forms of machine readable dictionaries and thesauri. Most resources organize lexical terms by alphabetical order. Roget's Thesaurus (Chapman, 1992) and WordNet (Miller, 1990) are probably best known for organizing lexical information in terms of semantic meanings.

Many researchers have attempted to extract computationally useful lexical information from machine readable dictionaries and convert this information into formal representations (Montemagni and Vanderwende, 1993). The syntactic information (i.e., parts of speech and morphologies) is routinely utilized for syntactic analysis. However, the use of semantic information from the generic lexical resources is still very limited. Generic lexical semantic resources have been used for some natural language processing tasks including word clustering, word sense disambiguation (Resnik, 1995; Yarowsky, 1992), and prepositional phrase attachment (Jensen and Binot, 1987; Harabagiu, 1996). However, the use of these resources in information extraction has not been fully investigated.

The purpose of information extraction (IE) is to identify and extract target information from a document and group this information into a coherent structural representation (i.e., templates). This task generally requires domain specific knowledge. The major concern is that the lexical semantic definitions given by the generic resources sometimes cannot meet the actual needs of the specific domain. The classification of words is sometimes too coarse and does not provide sufficient distinctions between words, and sometimes the classification is unnecessarily fine grained. Therefore, most IE systems have domain dependent lexicons (MUC6, 1995). No use is made of existing general lexical semantic resources by any of the MUC systems. IE system builders have tended to hand-craft resources for each application domain. This process is tedious, time consuming, and difficult to customize over different domains. Recently, the machine learning community has made several attempts to automatically learn the extraction rules from a relatively large set of training samples. This process involves manual annotation of the target information. This annotation usually requires considerable time and a certain amount of expertise (Riloff and Lehnert, 1993; Califf and Mooney, 1997; Soderland, 1999).

We have created a system that allows users to train the system on a small sample of documents. Then the system will generate and validate a set of rules based on the training information and WordNet. The validated rules will be applied to process new information. The system has three characteristics. First, the system generates rules automatically while traditional approach requires linguists to hand-craft extraction rules. Second, the system generates rules based on a small set of training samples and semantic knowledge from WordNet, while other automated learning approaches require a large number of annotations. Third, the system makes most linguistic features transparent such that users without strong linguistic expertise can customize the system for their domain of interest.

This paper describes the application of WordNet in such a system. After a brief overview of our earlier effort in applying WordNet for information extraction, the use of WordNet in rule generation and validation is described and evaluated. The results show that WordNet can help maximize effect of user training.

2. Information Extraction

Many information extraction systems rely heavily on hand-crafted, domain specific extraction rules. Other systems apply machine learning techniques on pre-annotated documents to automatically generate extraction rules. It is desirable to reduce the training effort and yet achieve reasonable results. We have explored WordNet for this purpose.

2.1. WordNet and Semantic Generalization

The conceptual hierarchy in WordNet offers a very desirable feature for natural language processing. Synonym and hypernym features naturally provide possibilities for

semantic generalization. However, the effectiveness of using WordNet in information extraction has been questioned. NYU's MUC-4 system used WordNet hierarchies for semantic classification. However, they ran into the problem of automated sense disambiguation because WordNet hierarchy is sense dependent. They concluded that "WordNet may be a good source of concepts, but that it will not be of net benefit unless manually reviewed with respect to a particular applications" (Grishman et al., 1992). The RAPIER system at the University of Texas tried to use WordNet to enhance the coverage of the extraction patterns learned from a comparably large training set, and they claimed that the impact of WordNet is not obvious.

We have applied several approaches to experiment the usefulness of WordNet in an information extraction system. In one of our earlier approaches, the system generates specific rules based on training examples (Bagga et al., 1997). A specific rule is made up of three entities. The first and the third entities are the target objects in the form of noun phrases. The second entity is the verb phrase or the preposition indicating the relationship between the two objects. WordNet is applied to generalize the first and third entities. We have conducted experiments to generalize rules to different degrees, (i.e., the system replaces the first and the third entities with the different concepts on the hypernym path). The results show that, within expectations, with the increase in the degree of generalization, precision tends to decrease while recall tends to increase. However, the best compromise between precision and recall as a function of generalization degree needs to be determined. Furthermore, the ability of the user to adjust precision at the cost of recall (or vice-versa) needs to be addressed.

Driven by those issues, we have designed a Generalization Tree Model that automatically learns the optimal level of semantic generalization based on user feedback (Chai and Biermann, 1997). The system first generalizes specific rules to the most general rules (i.e., replaces the first and the third entity with their respective top hypernym in WordNet hierarchy). Then it applies the most general rules back to the training set and extracts target objects and relations. It then asks the user to select the relevant target objects. Based on the user feedback, the system statistically determines which hypernyms on the path should be used to replace the first and the third in order to achieve the best performance.

To further simplify the training process, we have improved the system and designed rule generation and validation strategies to make better use of WordNet. In the following sections, we give a brief overview of the improved system and rule generation and validation strategies.

2.2. Overview of an Information Extraction System

The system consists of three major components that perform the basic sentence analysis: a Tokenizer, a Lexical Processor and a Partial Parser. The Tokenizer segments the input text into words and sentences. The Lexical Processor assigns words with syntactic information (such as morphological information) from CELEX databases (Baayen et al., 1993). The syntactic information is used in the Partial Parser which applies a finite state model to discover noun phrases, verb phrases and prepositions. In addition, the Lexical Processor employs a word sense disambiguation module to assign semantic information (i.e., senses from WordNet) to heads of phrases. In particular, a Semantic Classifier is used to identify special semantic types such as email and web addresses, file and directory names, dates, times, and dollar amounts, telephone numbers, and many others.

Figure 1: Screen shot of Training Interface

The system provides a graphical user interface for the user to train a small number of documents. As in Figure 1, depending on the position the cursor points to in the document, the interface shows the corresponding phrase identified by the Partial Parser. Associated with each phrase, the syntactic category, semantic type, headword, and potential senses (from WordNet) for the headword are also given. For example, in Figure 1, the current phrase is "Winston-Salem". It is a noun phrase (NG) and is identified as a city type. Since it's recognized as a special semantic type, the headword is the semantic type itself. Otherwise, the headword of a phrase is usually the base form of the last word in that phrase. During the training process, the user is required to identify some phrases as the information of interest (i.e., the target information). For example, in Figure 1, "Winston-Salem" is tagged as target information *LOCATION* type. Furthermore, the user needs to specify the correct sense for the headword. The training interface provides sense definitions to help the user make decisions. If no specification is given, the system assigns sense one, the most frequently used sense in WordNet to the headword. In addition, if the user is experienced, he/she can assist by specifying important phrases in the sentence that are crucial to the rule generation. If no such specification is given, the system will regard every phrase as an important phrase.

Based on the trained examples, the system automatically generates a set of extraction rules. First, the system creates all possible extractions rules following generation strategies. Then, it validates rules based on the limited training samples. Finally, it selects a subset of rules for future processing.

Important Phrases	Target	$(w, c, s, 0)$	$(w, c, s, 1)$	$(w, c, s, 2)$
Several projects	none	{undertaking,project,task,...}	{work}	{activity}
in	none	{in }		
Winston-Salem	LOCATION	{city, metropolis, ...}	{municipality}	{urban_area}
NC	LOCATION	{state,province}	{administrative_district,...}	{district,...}
experienced programmer analyst	POSITION	{analyst}	{expert}	{person,..}

Table 1: Important phrases from the training example and their corresponding concepts at different generalization degrees

2.3. Rule Generation and Validation

Rules are generated based on important phrases and user specified target information. For each important phrase in the training sentence, if the headword w exists in the WordNet, the synonym of w can be represented by $\{w, c, s, 0\}$ (where c is the syntactic category; s is the sense number and "0" stands for "synonym only"). By following the hypernym path d levels up, a more general concept represented by $\{w, c, s, d\}$ can be located. If the headword w is not in WordNet, then the concept includes only the word w itself and no generalization is available.

For example, as in Table 1, suppose a training sentence is: "Several projects are currently available in Winston-Salem, NC for experienced programmer analysts." After the user performs training on this sentence as described earlier, the system can generate corresponding concepts with different degrees of generalization from WordNet. Based on these concepts, the system follows several strategies to generate extraction rules for each type of target information.

2.3.1. Generation Strategies

Rules are pattern-action rules, with the left hand side specifying conditions and the right hand side as the action to extract certain type of target information. The left hand side of a rule consists of a conjunction of rule entities. Each rule entity is a subsumption function $S(X, \alpha)$. The subsumption function returns true if the headword of a phrase X is subsumed to the concept α. If all subsumption functions return true when matching this rule against a new sentence, then the right hand side action will take place to extract a corresponding phrase as a certain type of target information.

Strategy 1: Generate all possible rules with various number of rule entities.

For each type of target information, the system generates rules with one entity, two entities and up to k entities. The order of rule entities is preserved as the order of their corresponding important phrases in the training sentence.

For example, based on Table 1, for the target information *POSITION* type, the system can generate one rule with one entity on the left hand side as in equation 1. This rule implies that when a noun phrase has a headword subsumed to synonyms of "analyst", this noun phrase will be extracted as target information *POSITION*.

$$S(X_1, \{analyst, N, 1, 0\}) \longrightarrow FS(X_1, POSITION) \quad (1)$$

The system generate four rules with two entities on the left hand side as shown in Equation 2 to 5.

$$S(X_1, \{project, N, 1, 0\}) \wedge S(X_2, \{analyst, N, 1, 0\}) \longrightarrow FS(X_2, POSITION) \quad (2)$$

$$S(X_1, \{in, Prep, 1, 0\}) \wedge S(X_2, \{analyst, N, 1, 0\}) \longrightarrow FS(X_2, POSITION) \quad (3)$$

$$S(X_1, \{city, N, 1, 0\}) \wedge S(X_2, \{analyst, N, 1, 0\}) \longrightarrow FS(X_2, POSITION) \quad (4)$$

$$S(X_1, \{state, N, 2, 0\}) \wedge S(X_2, \{analyst, N, 1, 0\}) \longrightarrow FS(X_2, POSITION) \quad (5)$$

The system can also generate twelve rules with three rule entities. All rules generated by this strategy are referred to as combination rules and are potentially useful rules for extracting target information *POSITION*.

Rules with fewer entities are more general since they have less constraints. For some types of target information such as *SALARY*, one entity rule suffices. In the job advertisement domain, when a token is classified as some dollar amount semantic type, 95% of time it is the *SALARY* target information. However, for other types of target information such as *LOCATION*, since the geographic locations can be mentioned in different context, additional constraints will be necessary. By following this strategy, the system generates all possible rules so that it can learn from the training examples and decide the best number of rule entities dynamically.

Strategy 2: Generate all possible semantically generalized rules.

For each combination rule, the system replaces each concept in the subsumption function to a more general concept. For example,

$$S(X_1, \{project, N, 1, d_1(i)\}) \wedge S(X_2, \{analyst, N, 1, d_2(j)\}) \longrightarrow FS(X_2, POSITION) \quad (6)$$

where $\{project, N, 1, 0\}$ is replaced by a more general concept $\{project, N, 1, d_1(i)\}$ and $\{analyst, N, 1, 0\}$ is replaced by a more general concept $\{analyst, N, 1, d_2(j)\}$ (where $d_1(i) > 0$ and $d_2(j) > 0$) .

There is one rule corresponding to each combination of $d_1(i)$ and $d_2(j)$. If the distance from $\{project, N, 1, 0\}$ to its root concept in the hypernym hierarchy is D_1 and the distance from $\{analyst, N, 1, 0\}$ to its root concept is D_2, then totally $((D_1+1)*(D_2+1)-1)$ rules will be generated.

2.3.2. Validation Strategies

Validation strategies are applied to select useful rules from the large pool of potential rules.

Strategy 1: Select rules with precision_rate above certain threshold.

The system applies the generated rules back to the training examples. Based on the annotated information, the system computes *precision_rate* for each rule. The *precision_rate* of a rule is defined as the percentage of the correct information extracted by that rule. A threshold θ is predefined to control the process. If the precision is important to the user, the threshold should be set high (such as 0.9). If the recall is important, the threshold should be set relatively low (such as 0.6).

Strategy 2: Select semantically more generalized rules.

Suppose two rules r_1 and r_2 have the same number of entities, and the *precision_rate* for both rules is above the threshold. If every entity of r_1 corresponds to a more general concept than that of r_2, then the system will keep r_1 and discard r_2. The system will also sort the remaining rules to avoid repetitions.

By following these two validation strategies, the system selects a set of useful rules. Those rules will be applied to extract the target information from new documents.

When applying those rules on new documents, the system starts with rules with maximum number of entities. If there are some matches, the system extracts the target information as identified by the most number of matched rules. Otherwise, the system relaxes constraints by matching rules with fewer rule entities. This approach will first achieve the highest precision and then gradually increase recall without sacrificing precision.

3. Evaluation

We have tested the system on the *triangle.job* newsgroup where job advertisements are posted. The types of 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).

In our previous experiments (Chai et al., 1999), we randomly selected 24 documents for training and 40 documents for testing. Training articles were grouped into three training sets. The first training set contained 8 articles; the second contained 16 articles including ones in the first set; the third training set consisted of all 24 articles. In all of the experiments, the *precision_rate* threshold was set to 0.8 since it balanced recall and precision. In this particular trial, we have learned that rules generated by using WordNet enhanced the overall F-measurement about 10% when the training set only contained eight documents. In particular, those rules were very helpful in extracting certain facts. For *LOCATION* and *BENEFIT*, the performance was increased about 30% by those rules.

To continue our previous work, we conducted cross validation experiments. In each trial, from a total of 64 documents, we randomly selected 32 documents for training and the remaining 32 documents for testing. We further segmented 32 training documents into eight training sets with each containing 1, 2, 4, 8, 12, 16, 20, 24, 28, 32 documents. We conducted 10 trials and the performance reported here is the average performance.

In order to evaluate the use of WordNet, five types of rules were generated for comparison. Rules in the first category were generated without using WordNet (the performance is represented by *word only* curve). Rule entities only included headwords of important phrases from training examples. The rule match was only based on matching words and parts of speech. Rules in the second category were generated using WordNet synonyms (i.e., $max\ level\ =\ 0$ curve). Rule entities included synonyms of important headwords from training examples. The third type used WordNet hypernyms one level up in the hierarchy (i.e., $max\ level\ =\ 1$ curve). The fourth type included hypernyms two levels up (i.e., $max\ level\ =\ 2$ curve) and the fifth included hypernyms three levels up (i.e., $max\ level = 3$ curve).

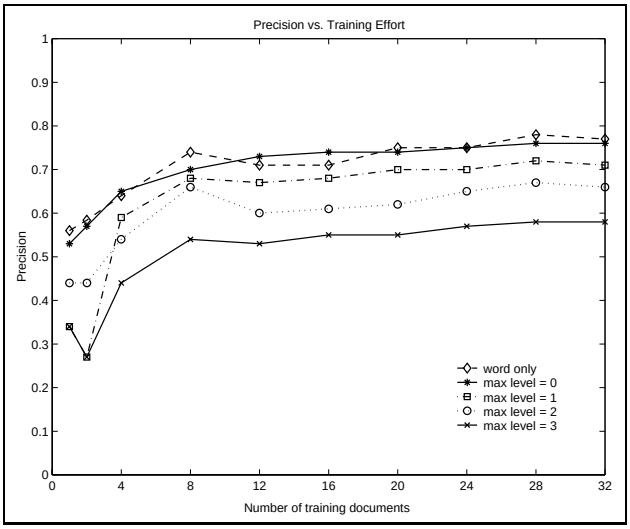

Figure 2: Rules with WordNet synonyms (denoted as $max\ level\ =\ 0$) achieve comparable precision with respect to rules without using WordNet (denoted as *word only*). They both outperform rules with WordNet hypernyms (denoted as $max\ level = 1, 2, 3$).

Figure 2 shows the average precision performance responding to the increase in training effort. Rules including WordNet synonyms achieve comparable precision with respect to rules generated from headwords only. They both

outperform rules including different levels of hypernyms. Because of the use of a threshold (0.8) in controlling the *precision_rate* of each rule, for all types of rules (except for the point at 1 and 2 training documents), precision goes up with the increase in the number of training documents.

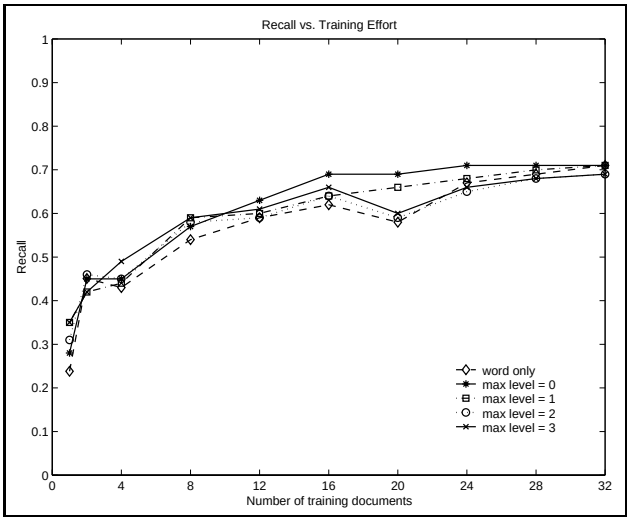

Figure 3: The use of WordNet improves recall. In particular, from 10-28 training documents, the use of synonyms (denoted as $max\ level = 0$) outperforms the rest. When the training set is 32 documents and larger, the impact of WordNet becomes insignificant.

Figure 3 shows the average recall performance responding to the increase in training effort. Within expectations, rules with WordNet synonyms perform better than rules without using WordNet. The recall difference increases from 2% to 11% with the increase in the number of training documents up to 20 documents. After that, the difference diminishes. Because vocabularies used in this domain are very limited, and variations of words can be directly learned after certain amount of training (for example, 32 documents).

Figure 4 shows the overall performance in response to an increase in training effort. In this example, WordNet synonyms achieve the best performance. With 20 documents, rules with synonyms achieve 72% F-measure, and rules without WordNet reach 65%. After this point, the difference again diminishes. With 32 training documents, the difference is negligible. This observation also confirms the claim that the impact of WordNet is not significant with the large training set.

4. Discussion

When customizing an information extraction system to a new domain, one can either hand-craft specific extraction rules or allow the system to learn extraction rules based on pre-annotated data. Both approaches require special expertise and are time consuming. Our system allows inexperienced users to train the system with minimum effort. The system automatically generates rules to maximize this effect of training by use of WordNet. The experiments demonstrate that WordNet can enhance the performance

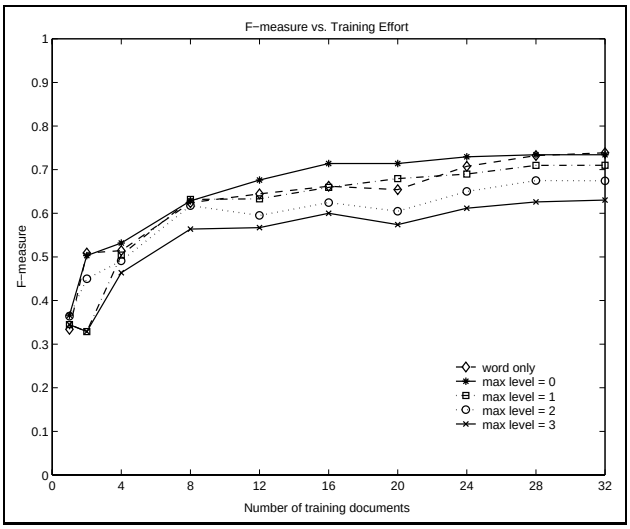

Figure 4: Training 16 documents with WordNet synonyms (denoted as $max\ level = 0$) achieves comparable overall performance as Training 28 documents without using WordNet (denoted as *word only*). This is an example of WordNet synonyms helping reduce training effort.

when the training data is small. As the training data becomes larger, the use of WordNet becomes less significant. However, since it is expensive to train a large amount of data, the use of WordNet can ease this process.

WordNet also has many limitations in information extraction. First of all, since WordNet hierarchy is sense dependent, word sense disambiguation is important. Our system includes a word sense disambiguation module. By allowing the user to select correct senses through the user interface, the system automatically generates word sense disambiguation rules (Chai and Biermann, 1999). Although we have observed that most words are used as sense one in WordNet and senses tend to remain the same in a specific domain, word sense disambiguation is still important with WordNet. Our experiments show that the overall system performance is enhanced by 8% when the word sense disambiguation module is applied. Word sense disambiguation certainly adds complexity to the use of WordNet; however, the sense training process is easier than the creation of a specific domain knowledge base, especially when definitions of senses are provided.

Another WordNet limitation lies in the fact that WordNet does not include proper nouns and domain specific terms. The target information is usually expressed in the form of a noun phrase. Some of these noun phrases are proper nouns (such as company names, person names, software names, and so on) which are not in WordNet. But most of them can be identified as special semantic types. The overall performance is directly affected by the accuracy in identifying those special semantic types. To make those special proper nouns connected to WordNet, our system creates a virtual link to make the concept of this special noun as a hyponym of the concept of its semantic type. For example, “IBM” is not in WordNet. The system first categorizes it as a company type, and then creates an “is-a” link

between {IBM} and {company}. Although WordNet does not provide semantic information to describe proper nouns and domain specific nouns, it does have a good coverage for verbs. To correctly extract target information, verbs are important since they convey certain relationships between different pieces of information. For example, in MUC6 domain, to extract the target information about who is leaving a certain position, suppose we have "resign" and "fire" as seed verbs which indicate this event. By only applying these seed verbs, 51.9% recall is achieved. By applying WordNet, 81.9% is achieved. The hypernyms and hyponyms of "resign" and "fire" are responsible for enhancing the performance.

Furthermore, in some cases, definitions of concepts in WordNet can not describe the intended use in the domain. For example, in the sentence "DCR Inc. is looking for accounting people", the word "people" has four senses in WordNet and they are all subsumed to the concept {group, grouping}. However, "people" in this sentence is referring to "professional" which is a kind of "person".

Based on above observations in using WordNet, it is desirable for WordNet to provide APIs for developers of information extraction systems to modify the database based on their needs. WordNet is a great resource and can be served as a core knowledge provider. The additional APIs can make the resource tailored toward specific needs.

5. Conclusion

WordNet can help maximize the effect of user effort in a trainable information extraction system. When the training data is small, the use of WordNet enhances the system performance by about 10%. However, when the training data is large, the impact of WordNet becomes insignificant. WordNet hypernyms have the risk of reducing performance because of over-generation. However, WordNet synonyms are beneficial to information extraction. By applying WordNet synonyms in rule generation and validation, our system generates useful rules based on the minimum training.

6. Acknowledgments

The author would like to thank Alan Biermann for insightful discussions and guidance; Jerry Hobbs for the finite state rules for the Partial Parser; Amit Bagga for developing the Tokenizer and the Semantic Classifier; and Robert McGough for his contributions to this manuscript.

7. References

Baayen, R. H., R. Piepenbrock, and H. van Rijn, 1993. *The CELEX Lexical Database on CD-ROM*. Philadelphia: Linguistic Data Consortium.

Bagga, A., J. Chai, and A. Biermann, 1997. The role of WordNet in the creation of a trainable message understanding system. *Proceedings of AAAI-97/IAAI-97*:941–948.

Califf, M. and R. Mooney, 1997. Relational learning of pattern-match rules for information extraction. *Proceedings of Computational Language Learning'97*.

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

Chai, J. and A. Biermann, 1999. The use of word sense disambiguation in information extraction. *Proceedings of Eleventh Conference on Innovative Applications of Artificial Intelligence (IAAI'99)*.

Chai, J., A. Biermann, and C. Guinn, 1999. Two dimensional generalization in information extraction. *Proceedings of Sixteenth National Conference on Artificial Intelligence (AAAI'99)*.

Chapman, L.R., 1992. *Roget's International Thesaurus*. Harper Collins, 5th edition.

Grishman, R., C. MacLeod, and J. Sterling, 1992. New York University PROTEUS system: MUC-4 test results and analysis. *Proceedings of the Fourth Message Understanding Conference (MUC-4)*:124–127.

Guthrie, L., J. Pustejovsky, Y. Wilks, and B. Slator, 1996. The role of lexicons in natural language processing. *Communications of ACM*.

Harabagiu, S., 1996. An application of WordNet to prepositional attachment. *Proceedings of ACL96*.

Jensen, K. and J. Binot, 1987. Disambiguating prepositional phrase attachments by using on-line dictionary definitions. *Computational Linguistics, 13(3)*:251–260.

Miller, G., 1990. WordNet: An on-line lexical database. *International Journal of Lexicography*:235–312.

Montemagni, S. and L. Vanderwende, 1993. Structural patterns versus string patterns for extracting semantic information from dictionaries. *Natural Language Processing: The PLNLP Approach*:149–159.

MUC6, 1995. *Proceedings of Sixth Message Understanding Conference*.

Resnik, P., 1995. Disambiguating noun grouping with respect to WordNet senses. *Third Workshop on Very Large Corpora*.

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

Soderland, S., 1999. Learning information extraction rules for semi-structured and free text. *Machine Learning Journal Special Issues on Natural Language Learning*, 34.

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*.