The Use of Lexical Semantics in Information Extraction *

Joyce Yue Chai **Alan W. Biermann**
Department of Computer Science
Box 90129, Duke University
Durham, NC 27708-0129
chai@cs.duke.edu awb@cs.duke.edu

Abstract

This paper presents a method for enabling users to specialize an information extraction system to satisfy their particular needs. The method allows the user to manually demonstrate the creation of semantic nodes and transitions while scanning a sample text article using a graphical user interface. On the basis of such examples, the system creates rules that translate text to semantic nets; then it generalizes these rules so that they can apply to a broad class of text instead of only the training articles. Finally, the generalized rules are used to scan large numbers of articles to extract the particular information targeted by the user. This paper concentrates on our design of the generalization mechanism which self modifies to precisely match the user's specification.

1 Introduction

Customizing information extraction systems across different domains has become an important issue in Natural Language Processing. Many research groups are making progress toward efficient customization, such as BBN (Weischedel, 1995), NYU (Grishman, 1995), SRI (Appelt et al., 1995), SRA (Krupka, 1995), MITRE (Aberdeen et al., 1995), UMass (Fisher et al., 1995)...etc. SRI developed a specification language called FASTSPEC that automatically translates regular productions written by the developer into finite state machines (Appelt et al., 1995). FASTSPEC makes the customization easier by avoiding the effort in enumerating all the possible ways of expressing the target information. The HASTEN system developed at SRA (Krupka, 1995) employs a graphical user interface that allows the user to create patterns by identifying the important concepts in the text, as well as the relationships between the concepts. Then the concepts are manually generalized to word classes before the patterns are applied to other texts from the domain.

We have built a trainable information extraction system that enables *any user* to adapt the system to different applications. The trainability of the system provides users the ability to identify the patterns for the information of interest. The training process is similar to the HASTEN system. However, instead of manual generalization as in HASTEN, our system automatically generalizes patterns by use of WordNet hierarchies. Automatic generalization of rules makes the customization process an easier one.

This paper describes the automated rule generalization method and the usage of WordNet (Miller, 1990) in our system. First, it introduces the idea of generalization; then it describes our Generalization Tree (GT) model based on the WordNet and illustrates how GT controls the degree of generalization according to the user's needs. Finally it demonstrates some preliminary results from the experiment of applying GT in our trainable information extraction system.

2 Lexical Acquisition

One way to achieve lexical acquisition is to use the existing repositories of lexical knowledge, such as knowledge base, dictionaries and thesauruses. The key issue is whether those repositories can be effectively applied for the computational purpose. Many researchers have taken steps toward successful extraction of computationally useful lexical information from machine readable dictionaries and convert it into formal representation (Montemagnia and Vanderwende, 1993) (Byrd et al., 1987) (Jensen and Binot, 1987). Sparck Jones's pioneering re-

*This work has been supported by a Fellowship from IBM Corporation.

search (Jones, 1985), done in early 1960, proposed a lexical representation by synonym list. Very close to that proposal, George Miller and colleagues at Princeton University constructed a large-scale resource for lexical information–WordNet.

The most useful feature of WordNet to Natural Language Processing community is the organization of lexical information in terms of word meanings, rather than word forms. It is organized by parts of speech–nouns, verbs, adjectives, and adverbs. Each entry in the WordNet is a concept represented by a list of synonyms (synset). The information is represented in the form of semantic networks. For instance, in the network for nouns, there are "part of", "is_a", "member of".... relationships between concepts. Philip Resnik has studied the lexical relationship by use of a WordNet taxonomy. He wrote that "...it is difficult to ground taxonomic representations such as WordNet in precise formal terms, the use of the WordNet taxonomy makes reasonably clear the nature of the relationships being represented. ..." (Resnik, 1993).

Some early work of applying WordNet for the lexical semantic acquisition can be found in NYU's MUC-4 system (Grishman et al., 1992), which used WordNet hierarchies for semantic classification. However, they ran into the problem of automated sense disambiguation because the WordNet hierarchy is sense dependent. Ralph Grishman and his group at NYU reached the conclusion 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 application" (Grishman et al., 1992). Other research concerns using WordNet senses to tag large corpus with the lexical semantics for automated word sense disambiguation (Ng, 1997) (Wiebe et al. , 1997)

3 Application of WordNet in the System

Our system contains three major processes which, respectively, address training, rule generalization, and the scanning of new information. WordNet is used in all three processes as shown in figure 1.

During the training process, each article is partially parsed and segmented into Noun Phrases, Verb Phrases and Prepositional Phrases. An IBM LanguageWare English Dictionary and Computing Term Dictionary, a Partial Parser, a Tokenizer and a Preprocessor are used in the parsing process. The Tokenizer and the Preprocessor are designed to identify some special categories such as e-mail address, phone number, state and city etc. The user, with

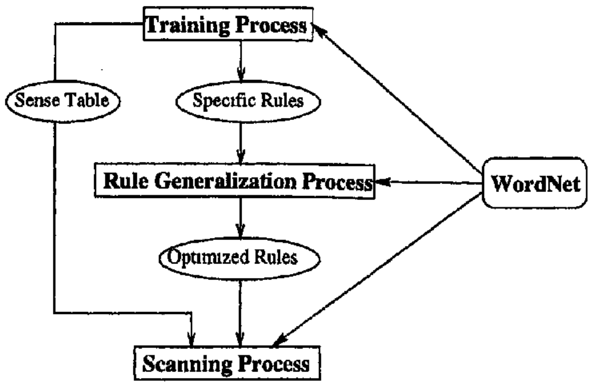


Figure 1: The Use of WordNet in the System

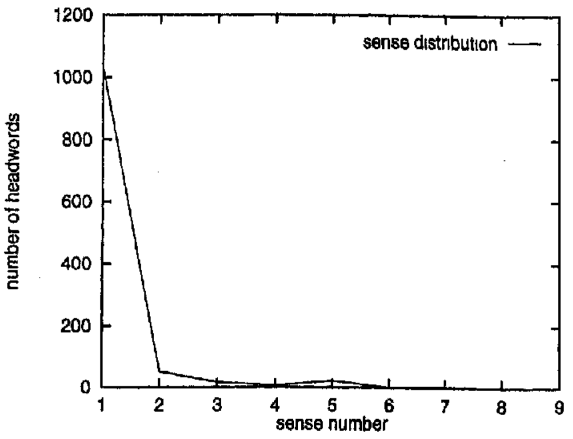


Figure 2: The Sense Distribution

the help of a graphical user interface(GUI) scans a parsed sample article and indicates a series of semantic net nodes and transitions that he or she would like to create to represent the information of interest. Specifically, the user designates those noun phrases in the article that are of interest and uses the interface commands to translate them into semantic net nodes. Furthermore, the user designates verb phrases and prepositions that relate the noun phrases and uses commands to translate them into semantic net transitions between nodes. In the process, the user indicates the desired translation of the specific information of interest into semantic net form that can easily be processed by the machine. When the user takes the action to create the semantic transitions, a Rule Generator keeps track of the user's moves and creates the rules automatically.

WordNet is used to provide the sense information during the training. For each headword in a noun/verb phrase, many senses are available in WordNet. We trained 24 articles with 1129 headwords from "triangle.job" domain, and found that 91.7% of headwords were used as sense one in WordNet. The sense distribution is shown in figure 2. Based on this observation, by default, the system assigns sense one to every headword, while providing the user the option to train the sense other than one. For example, "opening" often appears in the job advertisement domain. But instead of using the first sense as {*opening, gap*}, it uses the fourth sense as {*opportunity, chance*}. The user needs to train "opening" to be sense four during the training process. The Sense Table keeps the record of these headwords and their most frequently used senses (other than one).

Rules created from the training process are specific to the training articles and must be generalized before being applied on other articles in the domain. According to different requirements from the user, in the rule generalization process, a rule optimization engine, based on WordNet, generalizes the specific rules and forms a set of optimized rules for processing new information. This rule generalization process will be described in the later sections.

During the scanning of new information, with the help of a rule matching routine, the system applies the optimized rules on a large number of unseen articles from the domain. If headwords are not in the Sense Table, sense one in WordNet will be assigned; otherwise, the Sense Table provides them their most frequently used senses in the domain. The output of the system is a set of semantic transitions for each article that specifically extract information of interest to the user. Those transitions can then be used by a Postprocessor to fill templates, answer queries, or generate abstracts.

4 Rule Generalization

The Rule Generalization engine is crucial to the whole system because it makes the customizing process easier. The user only needs to train on a comparably small amount of data from the domain, and the system will automatically revise the rules to make them applicable for large amount of new information.

4.1 Rules

In a typical information extraction task, the most interesting part is the events and relationships holding among the events (Appelt et al., 1995). These relationships are usually specified by verbs and prepositions. Based on this observation, the left hand side (LHS) of our information extraction rules 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 or prepositional phrase indicating the relationship between the two objects. The right hand side (RHS) of the rule consists of the operations required to create a semantic transition–ADD_NODE, ADD_RELATION. ADD_NODE is to add an object in the transitions. ADD_RELATION is to add a relationship between two objects. The specific rule generated from the training process is shown in figure 3 rule 1.

Rule 1 in figure 3 is very specific, and it can be activated only by a sentence with the same pattern as "DCR Inc. is looking for C programmers . . . ". It will not be activated by other sentences such as "IBM Corporation seeks job candidates in Louisville, KY with HTML experience". Semantically speaking, these two sentences are very much alike. Both of them are about a company that seeks some kind of person. However, without generalization, the second sentence will not be processed. So the use of the specific rule is very limited.

In order to make the specific rules applicable to a large number of unseen articles in the domain, a comprehensive generalization mechanism is necessary. We are not only interested in the generalization itself, but also in a strategy to control the degree of generalization for various applications in different domains.

4.2 Generalization Scheme

The hierarchical organization of WordNet by word meanings (Miller, 1990) provides the opportunity for automated generalization. With the large amount of information in semantic classification and taxonomy provided in WordNet, many ways of incorporating WordNet semantic features with generalization are foreseeable. At this stage, we only concentrate on the Hypernym/Hyponym feature.

A hyponym is defined in (Miller et al., 1990a) as follows: " A noun X is said to be a hyponym of a noun Y if we can say that *X is a kind of Y*. This relation generates a hierarchical tree structure, i.e., a taxonomy. A hyponym anywhere in the hierarchy can be said to be "a kind of" all of its superordinateds. ..." If X is a hyponym of Y, then Y is a hypernym of X.

From the training process, the specific rules contain three entities on the LHS. An abstract specific rule is shown in rule 2 in figure 3. Each entity (sp) is a quadruple, in the form of (w, c, s, t), where w is the headword of the trained phrase; c is the part of the speech of the word; s is the sense number representing the meaning of w; t is the semantic type identified by the preprocessor for w.

For each $sp = (w, c, s, t)$, if w exists in WordNet,

1. **An Example of the Specific Rule:**

   [DCR Inc, NG, 1,company], [look_for, VG, 1, other_type], [programmer, NG, 1, other_type]
   $\longrightarrow$ ADD_NODE(DCR Inc.), ADD_NODE(programmer),
   ADD_RELATION(look_for, DCR Inc., programmer)

2. **An Abstract Specific Rule:**

   (w_1, c_1, s_1, t_1), (w_2, c_2, s_2, t_2),(w_3, c_3, s_3, t_3)
   $\longrightarrow$ ADD_NODE(w_1), ADD_NODE(w_2), ADD_RELATION(w_1, w_2, w_3)

3. **An Abstract Generalized Rule:**

   $(W_1, C_1, S_1, T_1) \in Generalize(sp_1, h_1)$, $(W_2, C_2, S_2, T_2) \in Generalize(sp_2, h_2)$,
   $(W_3, C_3, S_3, T_3) \in Generalize(sp_3, h_3)$
   $\longrightarrow$ ADD_NODE(W_1), ADD_NODE(W_3), ADD_RELATION(W_2,W_1, W_3)

4. **An Example of the Most General Rule:**

   $(W_1, C_1, S_1, T_1) \in \{group...\}$, $(W_2, C_2, S_2, T_2) \in \{look\_for, ...\}$, $(W_3, C_3, S_3, T_3) \in \{entity, ...\})$
   $\longrightarrow$ ADD_NODE(W_1), ADD_NODE(W_3), ADD_RELATION(W_2,W_1, W_3)

Figure 3: Sample Rules

then there is a corresponding synset in WordNet. The hyponym/hypernym hierarchical structure provides a way of locating the superordinate concepts of *sp*. By following additional Hypernymy, we will get more and more generalized concepts and eventually reach the most general concept, such as $\{entity\}$. Based on this scenario, for each concept, different degrees of generalization can be achieved by adjusting the distance between this concept and the most general concept in the WordNet hierarchy (Bagga et al., 1997). The function to accomplish this task is *Generalize(x,h)*, which returns a synset list h levels above the concept x in the hierarchy.

WordNet is an acyclic structure, which suggests that a synset might have more than one hypernym. However, This situation doesn't happen often. We tested on 150 randomly chosen articles from "triangle.job" newsgroup. Totally there were 12115 phrases including 1829 prepositions, 1173 phrases with headwords not in WordNet and 9113 phrases with headwords in WordNet. Within 9113 headwords, 722 headwords (7.9%), either themselves or their hypernym had more than one superordinate. Furthermore, 90% of 722 cases came from two superordinates of {*person, individual, someone, moral, human soul*}, which are {*life_form, organism, being, living thing*}, and {*causal agent, cause, causal agency*}. Certainly, in some cases, {*person...*} is a kind of {*causal agent...*}, but identifying it as hyponym of {*life_form...*} also makes the sense. Based on this scenario, for the sake of simplicity, the system selects the first superordinate if more than one are presented.

The process of generalizing rules consists of replacing each $sp = (w, c, s, t)$ in the specific rules by a more general superordinate synset from its hypernym hierarchy in WordNet by performing the $Generalize(s, h)$ function. The degree of generalization for rules varies with the variation of h in $Generalize(sp, h)$.

Rule 3 in figure 3 shows an abstract generalized rule. The $\in$ symbol signifies the subsumption relationship. Therefore, $a \in b$ signifies that a is subsumed by b, or, in WordNet terms, concept b is a superordinate concept of concept a. The generalized rule states that the RHS of the rule gets executed if

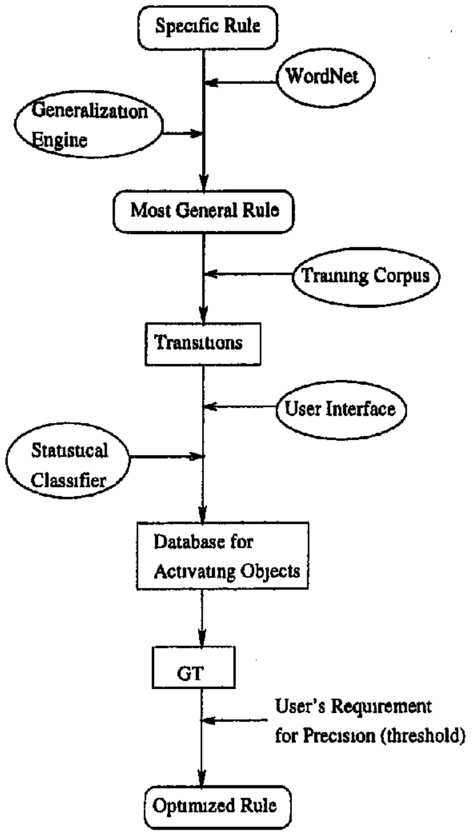


Figure 4: Rule Generalization Process

all of the following conditions hold:

- A sentence contains three phrases (not necessarily contiguous) with headwords W_1, W_2, and W_3.
- The quadruples corresponding to these headwords are (W_1, C_1, S_1, T_1), (W_2, C_2, S_2, T_2), and (W_3, C_3, S_3, T_3).
- The synsets, in WordNet, corresponding to the quadruples, are subsumed by *Generalize(sp_1, h_1)*, *Generalize(sp_2, h_2)*, and *Generalize(sp_3, h_3)* respectively.

5 Generalization Tree

The generalization degree is adjustable by the user. Rules with different degrees of generalization on their different constituents will have a different behavior when processing new texts. Within a particular rule, the user might expect one entity to be relatively specific and the other entity to be more general. For example, if a user is interested in finding all DCR Inc. related jobs, he/she might want to hold the first entity as specific as that in rule 1 in figure 3, and generalize the third entity. We have designed a Generalization Tree (GT) to control the generalization degree.

The rule generalization process with the help of GT is illustrated in figure 4. Each specific rule(as shown in rule 1 in figure 3) is generalized to its most general form(as shown in rule 4 in figure 3) by a generalization engine based on WordNet. Specifically, the generalization engine generalizes noun entities in the specific rule to their top hypernym in the hierarchies. The most general rule is applied again to the training corpus and some transitions are created. Some transitions are relevant, while others are not. Then the user employs our system to classify the created transitions as either acceptable or not. The statistical classifier calculates the relevancy_rate for each object, which will be described later. A database is maintained to keep the relevancy information for all the objects which activate the most general concept in the most general rule. This database is later automatically transformed to the GT structure. While maintaining the semantic relationships of objects as in WordNet, GTs collect the relevancy information of all activating objects and find the optimal level of generalization to fit the user's needs. The system will automatically adjust the generalization levels for each noun entity to match the desires of the user. The idea of this optimization process is to first keep recall as high as possible by applying the most general rules, then adjust the precision by tuning the rules based on the user's specific inputs.

5.1 An Example of GT

Suppose we apply rule 4 in figure 3 to the training corpus, and the entity three in the rule is activated by a set of objects shown in table 1. From a user interface and a statistical classifier, the relevancy_rate(*rel*) for each object can be calculated.

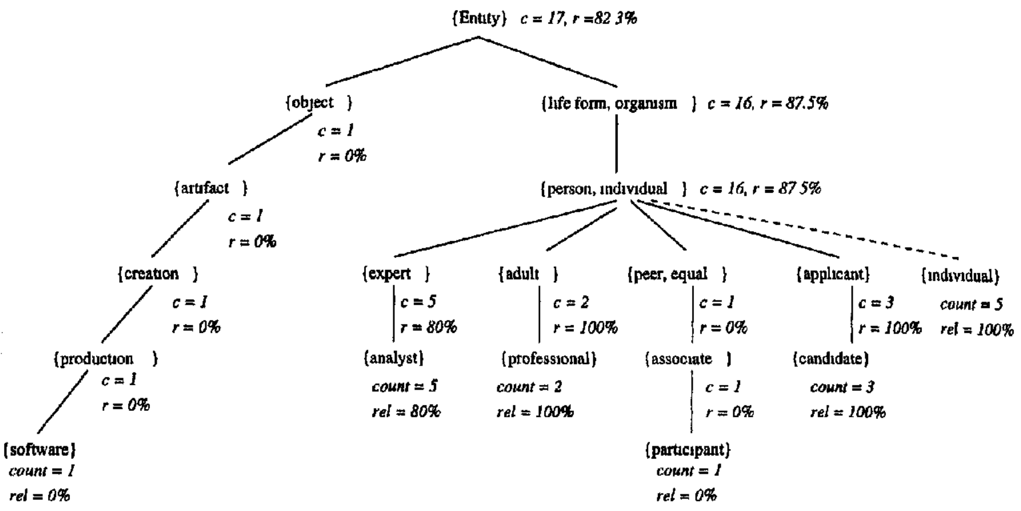
$$rel(obj) = \frac{count\ of\ obj\ being\ relevant}{total\ count\ of\ occurence\ of\ obj}$$

As shown in table 1, for example, $rel(\{analyst...\}) = 80\%$, which indicates that when $\{entity\}$ in the most general rule is activated by *analyst*, 80% of time it hits relevant information and 20% of time it hits irrelevant information. On the other hand, it suggests that if $\{entity\}$ is replaced by the concept $\{analyst...\}$, a roughly 80% precision could be achieved in extracting the relevant information. The corresponding GT for table 1 is shown in figure 5.

In GT, each activating object is the leaf node in the tree, with an edge to its immediate hypernym (parent). For each hypernym list in the database,

object	sense	hypernym list	depth	rel_rate	count
analyst	1	$\{analyst\} \Rightarrow \{expert\} \Rightarrow \{individual\} \Rightarrow \{life\ form\} \Rightarrow \{entity\}$	4	80%	5
candidate	2	$\{candidate\} \Rightarrow \{applicant\} \Rightarrow \{individual\} \Rightarrow \{life\ form\} \Rightarrow \{entity\}$	4	100%	3
individual	1	$\{individual\} \Rightarrow \{life\ form\} \Rightarrow \{entity\}$	2	100%	5
participant	1	....	5	0%	1
professional	1	...	4	100%	2
software	1	...	5	0%	1
...	...	...	...	...	...

Table 1: Sample Database for Objects Activating {*entity*}



{Entity} c = 17, r =82.3%
{object } c = 1 r = 0%
{life form, organism } c = 16, r = 87.5%
{artifact } c = 1 r = 0%
{person, individual } c = 16, r = 87.5%
{creation } c = 1 r = 0%
{expert } c = 5 r = 80%
{adult } c = 2 r = 100%
{peer, equal } c = 1 r = 0%
{applicant} c = 3 r = 100%
{individual} count = 5 rel = 100%
{production } c = 1 r = 0%
{analyst} count = 5 rel = 80%
{professional} count = 2 rel = 100%
{associate } c = 1 r = 0%
{candidate} count = 3 rel = 100%
{software} count = 1 rel = 0%
{participant} count = 1 rel = 0%

Figure 5: An Example of Generalization Tree

there is a corresponding path in GT. Besides the ordinary hard edge represented by the solid line, there is a soft edge represented by the dotted line. Concepts connected by the soft edge are the same concepts. The only difference between them is that the leaf node is the actual activating object, while the internal node is the hypernym for other activating objects. Hard edges and soft edges only have representational difference, as to the calculation, they are treated the same. Each node has two other fields *counts of occurrence* and *relevancy_rate*. For the leaf nodes, those fields can be filled from the database directly. For internal nodes, the instantiation of these fields depends on its hyponym (children) nodes. The calculation will be described later.

If the relevancy_rate for the root node {*entity*} is 82.3%, it indicates that, with the probability 82.3%, objects which activate {*entity*} are relevant. If the user is satisfied with this rate, then it's not necessary to alter the most general concept in the rule. If the user feels the estimated precision is too low, the system will go down the tree, and check the relevancy_rate in the next level. For example, if 87.5% is good enough, then the concept {*life form, organism...*} will substitute {*entity*} in the most general rule. If the precision is still too low, the system will go down the tree, find {*adult..*}, {*applicant..*}, and replace the concept {*entity*} in the most general rule with the union of these two concepts.

5.2 Generalization Tree Model

	object 1	relation	object 2
specific	x_0	y_0	z_0
...	....	....	....
more general	x_i		z_j
...	....	....	....
most general	x_n		z_m

Table 2: concepts in the rule

For the sake of simplicity, let's use x_i, y_i, z_i to represent the rule constituents– object one, relation, object two respectively. As shown in table 2, x_0, y_0, z_0 are the concepts from the specific rule. At the moment, we only consider the generalization on the objects. x_i and z_i are more general concepts than x_0 and z_0. x_i is the hypernym of x_{i-1} $(i \leq n)$; z_j is the hypernym of z_{j-1} $(j \leq m)$. x_n and z_m are the most general concepts for object one and object two respectively.

For each object concept, a corresponding GT is created. Let's suppose x_n is activated by q concepts $e_1^0, e_2^0, ...e_q^0$; the times of activation for each e_i^0 are represented by c_i. Since $e_i^0 (i \leq q)$ activates x_n, there exists a hypernym list $e_i^0 \Rightarrow e_i^1 \Rightarrow .... \Rightarrow x_n$ in WordNet, where e_i^j is the immediate hypernym of e_i^{j-1}. The system maintains a database of activation information as shown in table 3, and builds GT from this database automatically.

GT is an n-ary branching tree structure with the following properties:

- Each node represents a concept, and each edge represents the hypernym relationship between the concepts. If e_i is the immediate hypernym of e_j, then there is an edge between node e_i and e_j. e_i is one level above e_j in the tree.

- The root node x_n is the most general concept from the most general rule.

- The leaf nodes $e_1^0, e_2^0, ...e_q^0$ are the concepts which activate x_n. The internal nodes are the concepts e_j^i $(i \neq 0$ and $1 \leq j \leq q)$ from the hypernym paths for the activating concepts.

- Every leaf node e_i^0 has three fields–concept itself e_i^0 , *counts* and *relevancy_rate*, which can be obtained from the database:
  $counts(e_i^0) = c_i$
  $relevancy\_rate(e_i^0) = r_i$

- Every internal node e has three fields–concept itself e, *relevancy_rate* and *counts*(e).

  For an internal node e, if it has n hyponyms $e_0, ...e_n$ then:

$$counts(e) = \sum_{i=1}^{n} counts(e_i)$$

$$relevancy\_rate(e) = \sum_{i=1}^{n} P(e_i)*relevancy\_rate(e_i)$$

  where

$$P(e_i) = \frac{counts(e_i)}{counts(e)}$$

5.3 Searching GT

Depending on user's different needs, a threshold θ is pre-selected. The system will start from the root node, go down the tree, and find all the nodes e_i such that $relevancy\_rate(e_i) \geq \theta$. If a node *relevancy_rate* is higher than θ, its hyponym (children) nodes will be ignored. In this way, the system maintains a set of concepts whose *relevancy_rate* is higher than θ. By substituting x_n in the most general rule with this set of concepts, an optimized rule is created to meet the user's needs.

The searching algorithm is basically the breadth-first search as follows:

1. Initialize *Optimal-Concepts* to be empty set. Pre-select the threshold θ. If the user wants to get the relevant information and particularly cares about the precision, θ should be set high; if the user wants to extract as much as information possible and does not care about the precision, θ should be set low.

2. Starting from the root node x, perform the *Recursive-Search* algorithm, which is defined as the following:
   Recursive-Search(concept x)

```
{ if ($rel(x) \geq \theta$) {
    put x into Optimal-Concepts set;
    exit;
  }
  else {
      let m denote the number of children nodes of x;
      let $x_i$ denote the child of x ($0 < i \leq m$);
      for ( $i = 1; i \leq m; i++$ )
          Recursive-Search($x_i$);
  };
}
```

5.4 Experiment and Discussion

An experiment is conducted to test the applicability of GT in automatic information extraction. We trained our system on 24 articles from the *triangle.jobs* USENET newsgroups, and created 25 specific rules concerning the job *position/title* information. For example, in "DCR is looking for software engineer", *software engineer* is the position name. The specific rules then were generalized to their most general forms, and were applied again to the training set. After the user's selection of the relevant transitions, the system automatically generated a GT for each most general concept in the most general rule. We predefined the threshold to be 0.2, 0.4, 0.5, 0.6, 0.8, 0.9 and 1.0. Based on the different thresholds, the system generated different sets of optimized

activating objects	sense	counts	hypernym list	depth	relevancy_rate
e_1^0	s_1	c_1	$e_1^0 \Rightarrow e_1^1 \Rightarrow ... \Rightarrow x_n$	d_1	r_1
e_2^0	s_2	c_2	$e_2^0 \Rightarrow e_2^1 \Rightarrow ... \Rightarrow x_n$	d_2	r_2
....	...	...	....	....	...
e_q^0	s_q	c_q	$e_q^0 \Rightarrow e_q^1 \Rightarrow ... \Rightarrow x_n$	d_q	r_q

Table 3: database of activating concepts

rules. Those rules were then applied on 85 unseen articles from the domain.

The evaluation process consists of the following step: first, each unseen article is studied to see if there is *position/title* information presented in the article; second, the semantic transitions produced by the system are examined to see if they correctly extract the *position/title* information. Precision is the number of transitions created which containing *position/title* information out of the total number of transitions produced by the system; recall is the number of articles which have been correctly extracted *position/title* information out of the total number of articles with *position/title* information. The overall performance of recall and precision is defined by F-measurement (Chinchor, 1992), which is

$$\frac{(\beta^2 + 1.0) * P * R}{\beta^2 * P + R}$$

where P is precision, R is recall, $\beta = 1$ if precision and recall are equally important. The precision , recall and F-measurement curves with respect to the threshold for relevancy_rate are shown in figure 6. The detailed result is shown in table 4.

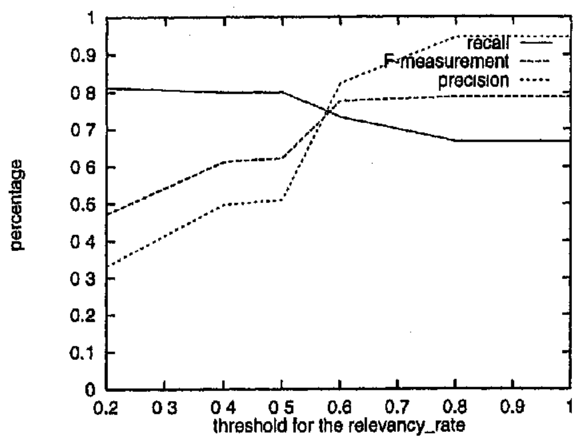


Figure 6: recall, precision, and F-measurement vs. threshold

The recall achieves the highest at 81.3% when $\theta = 0.2$. It gradually declines and reaches 66.7% when $\theta = 1.0$. As expected, the precision increases when θ goes up. It ranges from very low at 33.3% ($\theta = 0.2$) to very high at 94.6%($\theta = 1.0$). The overall performance F-measurement goes up from 47.2% to 78.7% when θ increases. The result is consistent with our expectation. When the threshold is high, more tuning of the rules needs to be done, and the system is expected to perform better.

Some problems were detected which prevent better performance of the system. The current domain is a newsgroup, where anyone can post anything which he/she believes is relevant to the newsgroup. It is inevitable that some typographical errors and some abbreviations occur in the articles. And the format of the article sometimes is unpredictable. The system performance is also hurt by the error in the partial parsing.

In the experiment, we found that WordNet has about 90% coverage of verbs and nouns in this domain. Most nouns not in WordNet are proper nouns, and in this domain, mostly are company names, software names. This problem is solved by our Preprocessor, which identifies the proper nouns to be several semantic types, such as company name, software name, city name, and so on. However Some important domain specific nouns may not exist in WordNet. It would be nice if WordNet could provide the friendly interface for users to add the new words and create the links for their own applications. As to computational purpose, WordNet is well developed. Finding hypernym, synonyms...etc is very efficient. Training senses at the training process solves the most problems of sense disambiguation. However, some problems still remain. For example, if "better" is not trained in the training process, then by default, it will be assigned sense one, which is a subtype of a person. The hypernym list of "better" with sense one is $\{better\} \Rightarrow \{superior\} \Rightarrow \{religion\} \Rightarrow \{Religionist\} \Rightarrow \{person\}$. But in the sentence "This position requires experience with 5.0 or better", "better" should be used as sense two as in the hypernym list $\{better\} \Rightarrow \{good, goodness\} \Rightarrow \{asset, plus\} \Rightarrow \{quality\} \Rightarrow \{attribute\} \Rightarrow \{abstraction\}$. Despite occasional sense disambiguation problem, generally, WordNet provides a good method to achieve generalization in

threshold	0.2	0.4	0.5	0.6	0.8	0.9	1.0
Precision	33.3%	49.7%	51.0%	82.4%	94.6%	94.6%	94.6%
Recall	81.3%	80.0%	80.0%	73.3%	66.7%	66.7%	66.7%
F-measurement	47.2%	61.3%	62.3%	77.6%	78.7%	78.7%	78.7%

Table 4: Precision/Recall/F-measurement wrt. threshold of relevancy_rate

this domain.

6 Conclusion and Future Work

This paper describes a rule generalization approach by using Generalization Tree and WordNet for information extraction. The rule generalization makes the customization process easier. The Generalization Tree algorithm provides a way to make the system adaptable to the user's needs. The idea of first achieving the highest recall with low precision, then adjusting precision by user's needs has been successful. We are currently studying how to enhance the system performance by further refining the generalization approach.

References

Aberdeen,John, John Burger, David Day, Lynette Hirschman, Patricia Robinson, and Marc Vilain 1995. MITRE: Description of the *ALEMBIC* System Used for MUC-6, *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, pp. 141-155, November 1995.

Appelt, Douglas E., Jerry R. Hobbs, John Bear, David Israel, Megumi Kameyama, Andy Kehler, David Martin, Karen Myers, and Mabry Tyson 1995. SRI International: Description of the FASTUS System Used for MUC-6, *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, pp. 237-248, November 1995.

Bagga, Amit, Joyce Y. Chai, and Alan W. Biermann 1997 The Role of WordNet in the Creation of a Trainable Message Understanding System, *To appear at The Innovative Applications of Artificial Intelligence Conference*, 1997.

Byrd, Roy, Nicoletta Calzolari, Martin Chodorow, Judith Klavans, and Mary Neff 1987 Tools and methods for computational linguistics, *Computational Linguistics, 13(3-4)*, pp. 219-240, 1987

Chai, Joyce Y.and Alan W. Biermann 1997 A WordNet Based Rule Generalization Engine For Meaning Extraction *Submitted to Tenth International Symposium On Methodologies For Intelligent Systems*, 1997.

Chinchor, Nancy 1992. MUC-4 Evaluation Metrics, *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, June 1992, San Mateo: Morgan Kaufmann.

Fisher, David, Stephen Soderland, Joseph McCarthy, Fangfang Feng and Wendy Lehnert. 1995. Description of the UMass System as Used for MUC-6, *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, pp. 127-140, November 1995.

Grishman, Ralph, Catherine Macleod, and John Sterling 1992. New York University Description of the Proteus System as Used for MUC-4, *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, pp. 223-241, June 1992.

Grishman, Ralph 1995. The NYU System for MUC-6 or Where's the Syntax? *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, pp. 167-175, November 1995.

Jensen, Karen, Jean-Louis Binot 1987. Disambiguating Prepositional Phrase Attachments by Using On-line Dictionary Definitions *Computational Linguistics, 13(3)* pp. 251-260, 1987.

Jones, Sparck 1985. Synonymy and Semantic Classification, Edinburgh University Press, 1985

Krupka, George R. 1995. Description of the SRA System as Used for MUC-6, *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, pp. 221-235, November 1995.

Miller, George A. 1990. Introduction to WordNet: An On-Line Lexical Database. *WordNet Manuals*, pp. 10-32, August 1993.

Miller, George A., et al. 1990a. *Five Papers on WordNet*, Cognitive Science Laboratory, Princeton University, No. 43, July 1990.

Montemagni, Simonetta, Lucy Vanderwende 1993 *Structural Patterns versus String Patterns for Extracting Semantic Information from Dictionaries*, Natural Language Processing: The PLNLP Approach, pp. 149-159, 1993

Ng, Hwee Tou 1997 Getting Serious about Word Sense Disambiguation, *ACL/SIGLEX Workshop on Tagging Text with Lexical Semantics*, pp. 1-7, Washington DC, April, 1997

Resnik, Philip 1993 Selection and Information: A Class Based Approach to Lexical Relationships, *Ph.D Dissertation, University of Pennsylvania*, 1993.

Weischedel, Ralph 1995. BBN: Description of the PLUM System as Used for MUC-6, *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, pp. 55-69, November 1995.

Wiebe, Janyce, Julie Maples, Lei Duan, and Rebecca Bruce 1997 Experience in WordNet Sense Tagging in Wall Street Journal, *ACL/SIGLEX Workshop on Tagging Text with Lexical Semantics*, pp. 1-7, Washington DC, April, 1997