Classifying Texts Using Relevancy Signatures

Ellen Riloff and Wendy Lehnert

Department of Computer Science
University of Massachusetts
Amherst, MA 01003
riloff@cs.umass.edu, lehnert@cs.umass.edu

Abstract

Text processing for complex domains such as terrorism is complicated by the difficulty of being able to reliably distinguish relevant and irrelevant texts. We have discovered a simple and effective filter, the *Relevancy Signatures Algorithm*, and demonstrated its performance in the domain of terrorist event descriptions. The Relevancy Signatures Algorithm is based on the natural language processing technique of selective concept extraction, and relies on text representations that reflect predictable patterns of linguistic context. This paper describes text classification experiments conducted in the domain of terrorism using the MUC-3 text corpus. A customized dictionary of about 6,000 words provides the lexical knowledge base needed to discriminate relevant texts, and the CIRCUS sentence analyzer generates relevancy signatures as an effortless side-effect of its normal sentence analysis. Although we suspect that the training base available to us from the MUC-3 corpus may not be large enough to provide optimal training, we were nevertheless able to attain relevancy discriminations for significant levels of recall (ranging from 11% to 47%) with 100% precision in half of our test runs.

Text Classification

Text classification is central to many information retrieval applications, as well as being relevant to message understanding applications in text analysis. To appreciate the importance and difficulty of this problem, consider the role that it played in the MUC-3 (The Third Message Understanding Conference) performance evaluation. Last year 15 text analysis systems attempted to extract information from news articles about terrorism (Lehnert & Sundheim 1991; Sundheim 1991). According to an extensive set of domain guidelines, roughly 50% of the texts in the MUC-3 development corpus did not contain legitimate information about terrorist activities. Articles that described rumours or lacked specific details were designated as irrelevant, as well as descriptions of specific events that targetted military personnel and installations (a terrorist event was defined to be one in which civilians or civilian locations were the apparent or accidental targets in an intentional act of violence). In order to achieve high-precision information extraction, the MUC-3 text analyzers had to differentiate relevant and irrelevant texts without human assistance. A system with a high rate of false positives would tend to generate output for irrelevant texts, and this behavior would show up in both the scores for overgeneration and spurious event counts. An analysis of the MUC-3 evaluation suggests that all of the MUC-3 systems experienced significant difficulty with relevant text classification (Krupka et al. 1991).

Although some texts will inevitably require in-depth natural language understanding capabilities in order to be correctly classified, we will demonstrate that skimming techniques can be used to identify subsets of a corpus that can be classified with very high levels of precision. Our algorithm automatically derives *relevancy signatures* from a training corpus using selective concept extraction techniques. These signatures are then used to recognize relevant texts with a high degree of accuracy.

Relevancy Discriminations

Terrorism is a complex domain, especially when it is combined with a complicated set of domain relevancy guidelines. Relevancy judgements in this domain are often difficult even for human readers. Many news articles go beyond the scope of the guidelines or fall into grey areas no matter how carefully the guidelines are constructed. Even so, human readers can reliably identify some subset of relevant texts in the terrorism domain with 100% precision, and often without reading these texts in their entirety. Text skimming techniques are therefore a

This research supported by the Office of Naval Research, under a University Research Initiative Grant, Contract #N00014-86-K-0764, NSF Presidential Young Investigators Award NSFIST-8351863, and the Advanced Research Projects Agency of the Department of Defense monitored by the Air Force Office of Scientific Research under Contract No. F49620-88-C-0058.

promising strategy for text classification as long as lower levels of recall[1] are acceptable. Although it might be unrealistic to try to classify all of the news articles in a corpus with a high degree of precision using anything less than a complete, in-depth natural language processing system, it is realistic to try to identify a subset of texts that can be accurately classified using relatively simple techniques.[2]

Intuitively, certain phrases seem to be very strong indicators of relevance for the terrorism domain. "X was assassinated" is very likely to be a reference to a terrorist event in which a civilian (politician, government leader, etc.) was killed. "X died" is a much weaker indicator of relevance because people often die in many ways that have nothing to do with terrorism. Linguistic expressions that predict relevance for a domain can be used to recognize and extract relevant texts from a large corpus. Identifying a reliable set of such expressions is an interesting problem and one that is addressed by relevancy feedback algorithms in information retrieval (Salton 1989).

Selective Concept Extraction using CIRCUS

Selective concept extraction is a sentence analysis technique that simulates the human ability to skim text and extract information in a selective manner. CIRCUS (Lehnert 1990) is a sentence analyzer designed to perform selective concept extraction in a robust manner. CIRCUS does not presume complete dictionary coverage for a particular domain, and does not rely on the application of a formal grammar for syntactic analysis. CIRCUS was the heart of the text analyzer underlying the UMass/MUC-3 system (Lehnert et al. 1991), and it provided us with the sentence analysis capabilities used in the experiments we are about to describe. [3]

The most important dictionary entries for CIRCUS are those that contain a concept node definition. Concept nodes provide the case frames that are used to structure CIRCUS output. If a sentence contains no concept node triggers, CIRCUS will produce no output for that sentence. One of the research goals stimulated by our participation in MUC-3 was to gain a better understanding of these concept nodes and the vocabulary items associated with them.

Our UMass/MUC-3 dictionary was hand-crafted specifically for MUC-3. A preliminary analysis of our MUC-3 dictionary indicated that we had roughly equal numbers of verbs and nouns operating as concept node triggers (131 verbs and 125 nouns). Other parts of speech also acted as concept node triggers, but to a lesser extent than verbs and nouns. Out of roughly 6000 dictionary entries, a total of 286 lexical items were associated with concept node definitions.

All concept node definitions contain a set of enablement conditions that must be met before the concept node can be considered valid. For example, if the lexical item "kill" is encountered in a sentence, a case frame associated with that item may be valid only if this instance of "kill" is operating as a verb in the sentence. Expectations for an agent and object will be useful for the verb "to kill" but not for a head noun as in "went in for the kill". Enablements are typically organized as conjunctions of conditions, and there is no restriction on what types of enablements can be used.

The enablement conditions for concept nodes effectively operate as filters that block further analysis when crucial sentence structures are not detected. If a filter is too strong, relevant information may be missed. If a filter is too weak, information may be extracted that is not valid. When sentence analysis fails due to poorly crafted enablement conditions, no other mechanisms can step in to override the consequences of that failure.

Relevancy Signatures

It is often the case that a single phrase will make a text relevant. For instance, a single reference to a kidnapping anywhere in a text generally signals relevance in the terrorism domain regardless of what else is said in the remainder of the article.[4] One implication of this fact is that it is not always necessary to analyze an entire text in order to accurately assess relevance. This property makes the technique of selective concept extraction particularly well-suited for text classification tasks.

We claim that specific linguistic expressions are reliable indicators of relevance for a particular domain. These expressions must be general enough to have broad applicability but specific enough to be consistently reliable

[1] **Recall** refers to the percentage of relevant texts that are correctly classified as relevant. **Precision** is the percentage of texts classified as relevant that actually *are* relevant. To illustrate the difference, imagine that you answer 3 out of 4 questions correctly on a true-or-false exam. Your recall rate is then 75%. Your precision, however, depends on how many of the questions you actually answered. If you only answered 3 of them, then your precision is 100%. But if you answered all 4 then your precision is only 75%.

[2] Many information retrieval tasks and message understanding applications are considered to be successful if low levels of recall are attained with high degrees of precision.

[3] The UMass/MUC-3 system posted the highest recall score and the highest combined scores for recall and precision of all the MUC-3 text analyzers (Sundheim 1991).

[4] In fact, there can be exceptions to any statement of this type. For example, an event that happened over 2 months ago was not considered to be relevant for MUC-3. Our approach assumes that these special cases are relatively infrequent and that key phrases can indicate relevance most of the time. Our technique will therefore produce weaker results under relevancy guidelines that detail special cases and exceptions if those conditions appear frequently in the target texts.

over large numbers of texts. For example, the word "dead" often appears in a variety of linguistic contexts such as "he was found dead", "leaving him dead", "left him dead", "they counted 15 dead", etc. Some of these expressions may provide stronger relevancy cues than others. For example, "<person> was found dead" is a strong relevancy cue since there is a good chance that the person was the victim of a terrorist crime, whereas "<number> dead" is a much weaker cue since it is often used in articles describing military episodes that are not terrorist in nature. Similarly, the word "casualties" by itself is not a strong relevancy cue since many articles discuss casualties in the context of military acts. But the expression "no casualties" *is* highly correlated with relevance since it often refers to civilians. We will refer to linguistic expressions that are strong relevancy cues as *relevancy signatures*.

In our system, these linguistic expressions are represented by ordered pairs of lexical items and concept nodes where the lexical item acts as a trigger for the concept node. For example, the pattern "was found dead" is represented by the pair ("dead", $found-dead-pass$) where dead is the key word that triggers the concept node $found-dead-pass$ which in turn activates enabling conditions that expect the passive form of the verb "found" to precede the word dead.

By taking advantage of the text corpus and answer keys used in MUC-3, we can automatically derive a set of relevancy signatures that will reliably predict the relevance of new texts. The following section describes the algorithm that derives a set of relevancy signatures from a training corpus and then uses those signatures to classify new texts.

The Relevancy Signatures Algorithm

MUC-3 provided its participants with a corpus of 1300 news articles for development purposes and two additional sets of 100 texts each that were made available for test runs (the TST1 and TST2 texts). All of the MUC-3 texts were supplied by the Foreign Broadcast Information Service and they were drawn from a variety of news sources including wire stories, transcripts of speeches, radio broadcasts, terrorist communiques, and interviews. The MUC-3 text corpus was supplemented by hand-coded case frame instantiations (answer keys) for each text in the corpus. The MUC-3 text corpus and answer keys therefore gave us access to 1500 texts and their correct relevancy classifications. For our experiments, we set aside a small portion of this corpus for testing purposes and dedicated the remaining texts to the training set. The training set was then used to derive a set of relevancy signatures.

The Relevancy Signatures Algorithm is fairly simple. Given a set of training texts, we parse each text using CIRCUS and save the concept nodes that are produced during the parse along with the lexical items that triggered those concept nodes. As we parse the training texts, we update two statistics for each word/concept node pair: [1] the number of times that the pair occurred in the training set (N), and [2] the number of times that it occurred in a relevant text (N_R). The ratio of N_R over N gives us a "reliability" measure. For example, .75 means that 75% of the instances (for that pair) appeared in relevant texts.

Using these statistics, we then extract a set of "reliable" lexical item/concept node pairs by choosing two values: a reliability threshold (R) and a minimum number of occurrences (M). The reliability threshold specifies the minimum reliability measure that is acceptable. For example, R=90 dictates that a pair must have a reliability measure greater than 90% in order to be considered reliable. The minimum number of occurrences parameter specifies a minimum number of times that the pair must have occurred in the training set. For example, M=4 dictates that there must be more than 4 occurrences of a pair for it to be considered reliable. This parameter is used to eliminate pairs that may have a very high reliability measure but have dubious statistical merit because they appeared only a few times in the entire training set. Once these parameters have been selected, we then identify all pairs that meet the above criteria. We will refer to these reliable word/concept node pairs as our set of relevancy signatures.

To illustrate, here are some relevancy signatures that were derived from the corpus using the parameter values, R=90 and M=10 along with some text samples that are recognized by these signatures:

("injured",$injury-1$)
the terrorists injured 5 people

("located",$location-pass-1$)
the banks were located

("occurred",$bomb-attack-2$)
an explosion occurred

("perpetrated",$perp-pass-1$)
the attack was perpetrated by ...

("placed",$loc-val-1$)
the terrorists placed a bomb ...

("placed",$loc-val-pass-1$)
a bomb was placed by ...

("planted",$loc-val-pass-1$)
a bomb was planted by ...

To classify a new text, we parse the text and save the concept nodes that are produced during the parse, along with the lexical items that triggered them. The text is therefore represented as a set of these lexical item/concept node pairs. We then consult our list of relevancy signatures to see if any of them are present in the current text. If we find one, the text is deemed to be relevant. If not, then the text is deemed to be irrelevant. It is important

to note that *it only takes one relevancy signature to classify a text as relevant.*

Experimental Results

To judge the effectiveness of the Relevancy Signatures Algorithm, we performed a variety of experiments. Since our algorithm derives relevancy signatures from a training set of texts, it is important that the training set be large enough to produce significant statistics. It is harder for a given word/concept node pair to occur than it is for only the word to occur, so many potentially useful pairings may not occur very often. At the same time, it is also important to have a large test set so we can feel confident that our results accurately represent the effectiveness of the algorithm. Because we were constrained by the relatively small size of the MUC-3 collection (1500 texts), balancing these two requirements was something of a problem. Dividing the MUC-3 corpus into 15 blocks of 100 texts each, we ran 15 preliminary experiments with each block using 1400 texts for training and the remaining 100 for testing. The results showed that we could achieve high levels of precision with non-trivial levels of recall. Of the 15 experiments, 7 test sets reached 80% precision with ≥ 70% recall, 10 sets hit 80% precision with ≥ 40% recall, and 12 sets achieved 80% precision with ≥ 25% recall. In addition, 7 of the test runs produced precision scores of 100% for recall levels > 10% and 5 test sets produced recall levels > 50% with precision over 85%.

Based on these experiments, we identified two blocks of 100 texts that gave us our best and our worst results. With these 200 texts in hand, we then trained once again on the remaining 1300 in order to obtain a uniform training base under which the remaining two test sets could be compared.

Figure 1 shows the performance of these two test sets based on the training set of 1300 texts. Each data point represents the results of the Relevancy Signatures Algorithm for a different combination of parameter values. We tested the reliability threshold at 70%, 75%, 80%, 85%, 90%, and 95% and varied the minimum number of occurrences from 0 to 19. As the data demonstrates, the results of the two test sets are clearly separated. Our best test results are associated with uniformly high levels of precision throughout (> 78%), while our worst test results ranged from 47% to 67% precision. These results indicate the full range of our performance: average performance would fall somewhere in between these two extremes.

Low reliability and low M thresholds produce strong recall (but weaker precision) for relevant texts while high reliability and high M thresholds produce strong precision (but weaker recall) for the relevant texts being retrieved. A high reliability threshold ensures that the algorithm uses only relevancy signatures that are very strongly correlated with relevant texts and a high minimum number of occurrences threshold ensures that it uses only relevancy signatures that have appeared with greater frequency. By adjusting these two parameter values, we can manipulate a recall/precision tradeoff.

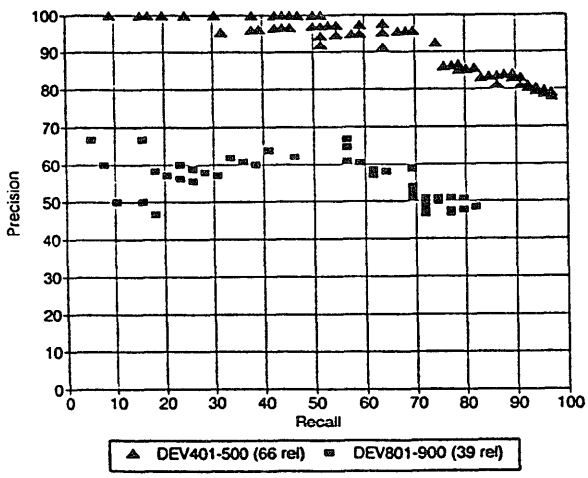

Figure 1: Relevancy Discriminations on Two Separate Test Sets Using Relevancy Signatures

However, a clear recall/precision tradeoff is evident only when the algorithm is retrieving statistically significant numbers of texts. We can see from the graph in Figure 1 that precision fluctuates dramatically for our worst test set when recall values are under 50%. At these lower recall values, the algorithm is retrieving such small numbers of texts (less than 20 for our worst test set) that gaining or losing a single text can have a significant impact on precision. Since our test sets contain only 100 texts each, statistical significance may not be reached until we approach fairly high recall values. With larger test sets we could expect to see somewhat more stable precision scores at lower recall levels because the number of texts being retrieved would be greater.

The percentage of relevant texts in a test set also plays a role in determining statistical significance. Each of the test sets contains a different number of relevant texts. For example, the best test set contains 66 relevant texts, whereas the worst test set contains only 39 relevant texts. The total percentage of relevant texts in the test corpus provides a baseline against which precision must be assessed. A constant algorithm that classifies all texts as relevant will always yield 100% recall with a precision level determined by this baseline percentage. If only 10% of the test corpus is relevant, the constant algorithm will show a 10% rate of precision. If 90% of the test corpus is relevant, the constant algorithm will achieve 90% precision. If we look at the graph in Figure 1 with this in mind, we find that a constant algorithm would yield 66% precision for the first test set but only 39% for the second test set. From this vantage point, we can see that the Relevancy Signatures Algorithm performs substantially better than the constant algorithm on both test sets.

It was interesting to see how much variance we got across the different test sets. Several other factors may have contributed to this. For one, the corpus is not a randomly ordered collection of texts. The MUC-3 articles were often ordered by date so it is not uncommon to find

sequences of articles that describe the same event. One block of texts may contain several articles about a specific kidnapping event while a different block will not contain any articles about kidnappings. Second, the quality of the answer keys is not consistent across the corpus. During the course of MUC-3, each participating site was responsible for encoding the answer keys for different parts of the corpus. Although some cross-checking was done, the quality of the encoding is not consistent across the corpus.[5] The quality of the answer keys can affect both training and testing.

The relatively small size of our training set was undoubtedly a limiting factor since many linguistic expressions appeared only a few times throughout the entire corpus. This has two ramifications for our algorithm: (1) many infrequent expressions are never considered as relevancy signatures because the minimum number of occurrences parameter prohibits them, and (2) expressions that occur with low frequencies will yield less reliable statistics. Having run experiments with smaller training sets, we have seen our results show marked improvement as the training set grows. We expect that this trend would continue for training sets greater than 1400, but corpus limitations have restricted us in that regard.

The Augmented Relevancy Signatures Algorithm

Relevancy signatures were motivated by our observation that human readers can reliably identify many relevant texts merely by skimming the texts for domain-specific cues. These quick relevancy judgements require two steps: (1) recognizing an expression that is highly relevant to the given domain, e.g. "were killed" in the domain of terrorism, and (2) verifying that the context surrounding the expression is consistent with the relevancy guidelines for the domain, e.g. "5 soldiers were killed by guerrillas" is not consistent with the terrorism domain since victims of terrorist acts must be civilians. The Relevancy Signatures Algorithm simulates the first step in this process but can misclassify texts when the surrounding context contains additional information that makes the text irrelevant.

In particular, relevancy signatures do not take advantage of the slot fillers in the concept nodes. For example, consider two similar sentences: (a) "a civilian was killed by guerrillas" and (b) "a soldier was killed by guerrillas". Both sentences are represented by the same relevancy signature: (killed, $murder-pass-1$) even though sentence (a) describes a terrorist event and sentence (b) does not. To address this problem, we developed a variation of the Relevancy Signatures Algorithm that augments the relevancy signatures with slot filler information. While relevancy signatures classify texts based upon the presence of case frames, augmented relevancy signatures classify texts on the basis of case frame instantiations. The algorithm for deriving and using augmented relevancy signatures is described below.

Given a set of training texts, we parse each text and save the concept nodes that are generated. For each slot in each concept node, we collect reliability statistics for triples consisting of the concept node type, the slot name, and the semantic feature of the filler.[6] For example, consider the sentence: "The mayor was murdered." The word "murdered" triggers a murder concept node that contains "the mayor" in its *victim* slot. This concept node instantiation yields the slot triple: (murder, victim, ws-government-official). We then extract a set of "reliable" slot triples by choosing two values: a reliability threshold R_{slot} and a minimum number of occurrences threshold M_{slot}. These parameters are analogous to the relevancy signature thresholds.

To classify a new text, we parse the text and save the concept nodes that are produced during the parse, along with the words that triggered them. For each concept node, we generate a (triggering word, concept node) pair and a set of slot triples. If the (triggering word, concept node) pair is in our list of relevancy signatures, **and** the concept node contains a reliable slot triple then we classify the text as relevant. Intuitively, a text is classified as relevant only if it contains a strong relevancy cue **and** the concept node enabled by this cue contains at least one slot filler that is also highly correlated with relevance.

More Experimental Results

We compared the performance of the augmented relevancy signatures with the original Relevancy Signatures Algorithm in order to measure the impact of the slot filler data. Figure 2 shows the results produced by the augmented relevancy signatures on the same two test sets that we had isolated for our original experiments, after training on the remaining 1300 texts. Each data point represents a different combination of parameter values.

This graph clearly shows that the augmented relevancy signatures perform better than the original relevancy signatures on these two test sets. The most striking difference is the improved precision obtained for DEV 801-900. There are two important things to notice about Figure 2. First, we are able to obtain extremely high precision at low recall values, e.g., 8% recall with 100% precision and 23% recall with 90% precision. Relevancy signatures alone do not achieve precision greater than 67%

[5] During the course of this research, we found that about 4% of the irrelevant texts in the MUC-3 development corpus were miscategorized. These errors were uncovered by spot checks: no systematic effort was made to review all the irrelevant texts. We therefore suspect that the actual error rate is probably much higher.

[6] Since slot fillers can have multiple semantic features, we create one triple for each feature. For example, if a murder concept node contains a victim with semantic features *ws-human & ws-military* then we create two triples: (murder, victim, *ws-human*) and (murder, victim, *ws-military*).

for this test set at any recall level. Second, although there is a very scattered distribution of data points at the lower recall end, we see consistently better precision coupled with the higher recall values. This trend suggests that the augmented relevancy signatures perform better than the original relevancy signatures when they are working with statistically significant numbers of texts.

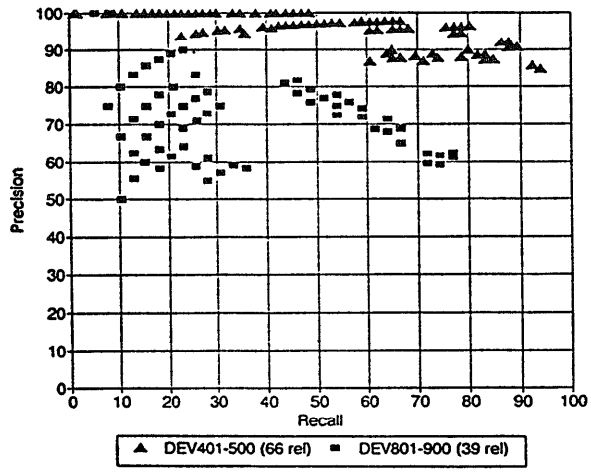

Figure 2: Relevancy Discriminations on Two Separate Test Sets Using Augmented Relevancy Signatures

Noting that the relevancy signatures demonstrated extremely strong performance on DEV 401-500, it is reassuring to see that the augmented relevancy signatures achieve comparable results. The highest recall level obtained with extremely high precision by the original relevancy signatures was 70% with 96% precision. The augmented relevancy signatures achieved significantly higher recall with the same precision, 80% recall with 96% precision. This suggests that augmented relevancy signatures can also achieve considerably greater levels of recall with high precision than relevancy signatures alone.

Conclusions

The Relevancy Signatures Algorithm was inspired by the fact that human readers are capable of scanning a collection of texts, and reliably identifying a subset of those texts that are relevant to a given domain. More importantly, this classification can be accomplished by fast text skimming: the reader hits on a key sentence and a determination of relevancy is made. This method is not adequate if one's goal is to identify *all* possible relevant texts, but text skimming can be very reliable when a proper subset of relevant texts is sufficient. We designed the Relevancy Signatures Algorithm in an effort to simulate this process.

In fact, the Relevancy Signatures Algorithm has an advantage over humans insofar as it can automatically derive domain specifications from a set of training texts. While humans rely on domain knowledge, explicit domain guidelines, and general world knowledge to identify relevant texts, the Relevancy Signatures Algorithm requires no explicit domain specification. Given a corpus of texts tagged for domain relevancy, an appropriate dictionary, and suitable natural language processing capabilities, reliable relevancy indicators are extracted from the corpus as a simple side effect of natural language analysis. Once this training base has been obtained, no additional capabilities are needed to classify a new text.

It follows that the Relevancy Signatures Algorithm avoids the knowledge-engineering bottleneck associated with many text analysis systems. As a result, this algorithm can be easily ported to new domains and is trivial to scale-up. With large online text corpora becoming increasingly available to natural language researchers, we have an opportunity to explore operational alternatives to hand-coded knowledge bases and rule bases. As we have demonstrated, natural language processing capabilities can produce domain signatures for representative text corpora that support high-precision text classification.

Acknowledgements

We would like to thank David Fisher for ongoing technical support that enabled preliminary experiments with the UMass/MUC-3 system on a Macintosh platform.

References

1. Krupka, G., Iwanska, L., Jacobs, P., and Rau, L. 1991. GE NLToolset: MUC-3 Test Results and Analysis. In Proceedings of the Third Message Understanding Conference, 60-68. San Mateo, CA. Morgan Kaufmann.

2. Lehnert, W.G. 1990. Symbolic/Subsymbolic Sentence Analysis: Exploiting the Best of Two Worlds. In *Advances in Connectionist and Neural Computation Theory*. (Eds: J. Pollack and J. Barnden), 135-164. Norwood, NJ. Ablex Publishing.

3. Lehnert, W.G., Cardie, C., Fisher, D., Riloff, E., and Williams, R. 1991. The CIRCUS system as used in MUC-3, COINS Technical Report 91-59. Department of Computer and Information Science, University of Massachusetts at Amherst.

4. Lehnert, W.G. and Sundheim, B. 1991. A Performance Evaluation of Text Analysis Technologies. *AI Magazine*, vol 12; no.3, pp. 81-94.

5. Salton, G. 1989. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. Reading, MA. Addison-Wesley Publishing Company, Inc.

6. Sundheim, B. 1991. (ed.) Proceedings of the Third Message Understanding Conference. San Mateo, CA. Morgan Kaufmann.