In *Connectionist, Statistical, and Symbolic Approaches to Learning for Natural Language Processing*, S. Wermter, E. Riloff, and G. Scheler, eds. 1996. Springer-Verlag, Berlin. pp. 275-289

Using Learned Extraction Patterns for Text Classification

Ellen Riloff

Department of Computer Science
University of Utah
Salt Lake City, UT 84112, USA
riloff@cs.utah.edu

Abstract. A major knowledge-engineering bottleneck for information extraction systems is the process of constructing an appropriate dictionary of extraction patterns. AutoSlog is a dictionary construction system that has been shown to substantially reduce the time required for knowledge engineering by learning extraction patterns automatically. However, an open question was whether these extraction patterns were useful for tasks other than information extraction. We describe a series of experiments that show how the extraction patterns learned by AutoSlog can be used for text classification. Three dictionaries produced by AutoSlog for different domains performed well in our text classification experiments.

1 Introduction

Many researchers in natural language processing have turned their attention recently to a problem called *information extraction* (IE). Information extraction is a natural language processing task that involves extracting predefined types of information from text. Information extraction is essentially a form of text-skimming because only the portions of a text that are relevant to a given domain need to be understood. However, it is crucial for an IE system to have a knowledge base of concepts that provides it with good coverage of the domain.

The challenge for information extraction researchers is to develop methods for acquiring the necessary dictionaries and knowledge bases automatically. To this end, we have developed a system called AutoSlog [Riloff, 1996; Riloff, 1993] that automatically constructs dictionaries of domain-specific extraction patterns. In the domain of Latin American terrorism, a dictionary created by AutoSlog achieved 98% of the performance of a hand-crafted dictionary that took approximately 1500 person-hours to build. We have also used AutoSlog to create dictionaries of extraction patterns for a joint ventures domain and a microelectronics domain.

One of the questions raised by the growing interest in information extraction is whether the methods and resources developed for information extraction will

be useful for other natural language processing tasks. We decided to investigate this issue by applying information extraction dictionaries and techniques to the problem of text classification. In a series of experiments, we used an information extraction system and dictionaries produced by AutoSlog to generate *relevancy signatures* [Riloff and Lehnert, 1994] for automatic text classification. The heart of the information extraction system is its dictionary of extraction patterns, so these experiments also served to demonstrate that the AutoSlog dictionaries were useful for making important domain discriminations.

In Section 2, we overview information extraction and the CIRCUS sentence analyzer that was used in these experiments. In Section 3, we describe the AutoSlog dictionary construction system that automatically creates dictionaries of extraction patterns using an annotated training corpus. In Section 4, we describe the relevancy signatures algorithm for text classification and present the results of the text classification experiments. Finally, we discuss related work in machine learning and automated dictionary construction and draw some conclusions.

2 A Brief Introduction to Information Extraction

Information extraction is a natural language processing task that involves automatically extracting predefined types of information from text. In contrast to in-depth understanding, information extraction systems focus only on portions of text that are relevant to a specific domain (e.g., see [Jacobs and Rau, 1990; Lehnert and Sundheim, 1991]). For example, an information extraction system designed for a terrorism domain might extract the names of perpetrators, victims, physical targets, and weapons involved in a terrorist attack. Or an information extraction system designed for a joint ventures domain might extract the names of companies involved in a joint venture and products, facilities, or people associated with those companies.

CIRCUS [Lehnert, 1991] is a conceptual sentence analyzer that performs information extraction. CIRCUS uses a dictionary of *concept nodes* to recognize domain-specific patterns and expressions and to extract relevant information. A concept node is essentially a case frame that is activated by specific linguistic expressions and extracts information from the surrounding context. To illustrate, Figure 1 shows a simple sentence and the resulting instantiated concept node produced by CIRCUS. The concept node $murder-passive$ is activated by passive forms of the verb "murdered", such as "were murdered" in the sample sentence. It then extracts the subject of the verb as the murder victim and the object of the preposition "by" as the perpetrator. In Figure 1, the "three peasants" are extracted as murder victims, and the "guerrillas" are extracted as perpetrators. A similar concept node called $murder-active$ is triggered by active forms of the verb "murdered", and extracts the subject as the perpetrator and the direct object as the victim.

A sentence may activate multiple concept nodes if more than one relevant expression is found, or a sentence may not activate any concept nodes if no relevant expressions are found. All of the information extraction happens through

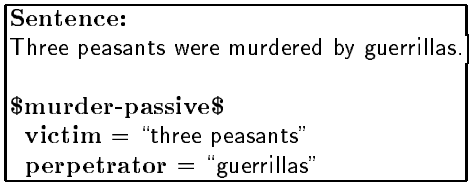

Fig. 1. An instantiated concept node

concept nodes, so it is essential for CIRCUS to have a concept node dictionary that provides good coverage of the domain.

3 AutoSlog: Learning Extraction Patterns

CIRCUS was the central component of the UMass/MUC-3[1] system [Lehnert *et al.*, 1991] developed at the University of Massachusetts. The concept node dictionary used by the UMass/MUC-3 system was built by hand. The dictionary performed well[2], but took approximately 1500 person-hours to build [Lehnert *et al.*, 1992]. The dictionary construction process was a major knowledge-engineering bottleneck that limited the scalability and portability of the system.

Subsequently, we developed a system called AutoSlog [Riloff, 1996; Riloff, 1993] that learns domain-specific extraction patterns from an annotated training corpus. As input, AutoSlog needs a text and a set of tagged noun phrases that represent the information that should have been extracted from the text. Each noun phrase needs to be tagged with its semantic type and the event type with which it is associated. Figure 2 shows a sample sentence and annotated noun phrases.

For each tagged noun phrase, AutoSlog proposes a linguistic pattern that is capable of extracting the noun phrase in an appropriate context. Intuitively, AutoSlog tries to find a pattern that identifies the role of the noun phrase in a relevant event or context. For example, consider the sentence:

(a) John Smith was murdered by armed men.

Suppose "John Smith" is tagged as a murder victim and the phrase "armed men" is tagged as the perpetrator of a murder. AutoSlog will propose that the pattern "<X> was murdered" identifies X as the victim of a murder and should therefore be used to extract John Smith. Since the pattern itself is general, it can be used to extract the names of other murder victims in future texts. Similarly, AutoSlog will propose that the pattern "was murdered by <Y>" identifies Y as a perpetrator and should therefore be used to extract the armed men. Again, the pattern is general so it can be used to extract new perpetrators in the future.

[1] The MUCs refer to the message understanding conferences, which are competitive performance evaluations of information extractions systems.

[2] The UMass/MUC-3 system earned the highest combined recall and precision scores of the 15 sites that participated in MUC-3 [MUC-3 Proceedings, 1991].

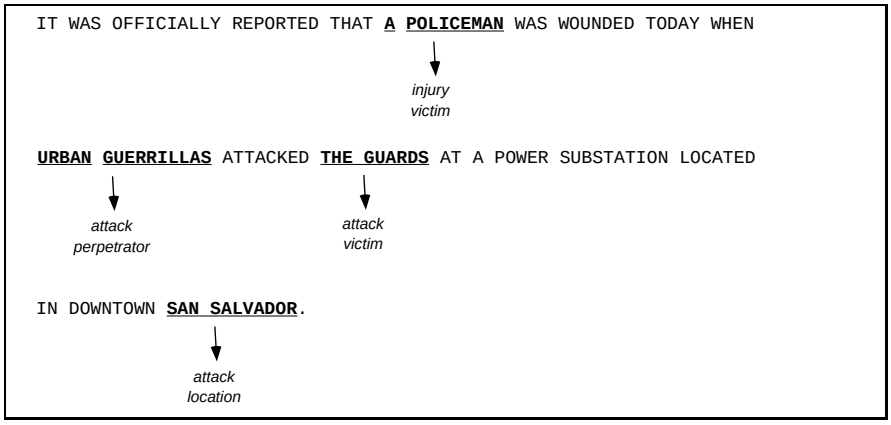

Fig. 2. Example text annotations for AutoSlog

The core of AutoSlog is a simple set of linguistic rules that generate the extraction patterns. Each rule depends solely on the syntactic location of the targeted noun phrase and the surrounding context. The original set of AutoSlog rules[3] is shown in Figure 3. The rules are divided into three sets, depending upon whether the targeted noun phrase was the subject, direct object, or prepositional phrase of a clause. If the noun phrase was the subject, then AutoSlog hypothesizes that the verb determines the role of the noun phrase and should be used as the basis for the pattern. The exact form of the pattern mirrors the syntactic context surrounding the noun phrase.

For example, in sentence (a) AutoSlog determines that "John Smith" is the subject of the sentence so the <subject> rules are activated. Since "murdered" is in the passive voice, the **<subject> passive-verb** rule matches the surrounding context. The rule fires and instantiates its general pattern with the specific verb found in the sentence. The resulting extraction pattern represents expressions such as "X was murdered", "X and Y were murdered", and "X has been murdered." The process is similar for the other two categories of rules. If CIRCUS determines that the targeted noun phrase is a direct object, then those rules are activated. In most cases, the verb is used as the basis for the pattern because the verb usually identifies the conceptual role of the noun phrase. If CIRCUS determines that the targeted noun phrase is in a prepositional phrase, then a simple pp-attachment algorithm is used to attach it to a preceding verb or noun. That verb or noun is then used to create a new extraction pattern.

Figure 4 shows the flow of control in the AutoSlog system. As input, AutoSlog needs texts and tagged noun phrases. For the experiments described here, we used the MUC answer keys as input. It is important to note that the answer keys contain a lot of additional information that AutoSlog does not need and did not use. We used the MUC answer keys because they were readily available,

[3] A slightly different set of rules was used in the joint ventures and microelectronics domains. See [Riloff, 1996] for details.

Linguistic Pattern	Example
<subject> active-verb	<perpetrator> bombed
<subject> passive-verb	<victim> was murdered
<subject> verb infinitive	<perpetrator> attempted to kill
<subject> auxiliary noun	<victim> was victim
active-verb <direct-object>	bombed <target>
passive-verb <direct-object>	killed <victim>
infinitive <direct-object>	to kill <victim>
verb infinitive <direct-object>	threatened to attack <target>
gerund <direct-object>	killing <victim>
noun auxiliary <direct-object>	fatality was <victim>
noun preposition <noun-phrase>	bomb against <target>
active-verb preposition <noun-phrase>	killed with <instrument>
passive-verb preposition <noun-phrase>	was aimed at <target>

Fig. 3. AutoSlog rules and examples in the terrorism domain

but an annotated corpus would have been sufficient.

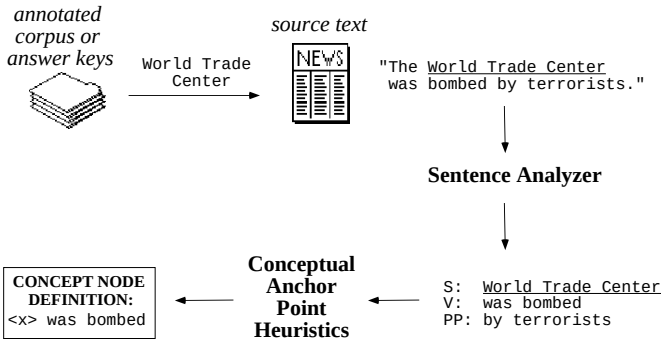

Fig. 4. AutoSlog flowchart

Given the annotated corpus (or answer keys), AutoSlog processes one noun phrase at a time. Given a tagged noun phrase, AutoSlog first finds the sentence from which the noun phrase originated. If there is no pointer from the noun phrase back to the original source text, then AutoSlog uses the first sentence in the text that contains the noun phrase. The sentence is then pushed through CIRCUS, which breaks it up into clauses and identifies the major syntactic constituents of each clause (subject, verb, direct object, and prepositional phrases). The appropriate AutoSlog rules are activated depending upon the syntactic location of the noun phrase. The rules are evaluated in order, and the first rule to recognize its pattern will generate the extraction pattern for the noun phrase.

Figure 5 shows an example of a concept node generated by AutoSlog for the terrorism domain. The targeted noun phrase is "guerrillas", which is tagged as a perpetrator. AutoSlog searches for the first sentence containing the word "guerrillas", shown at the top of Figure 5, and sends it through CIRCUS. CIRCUS determines that the guerrillas are the subject of the first clause in the sentence so AutoSlog's <subject> rules are activated. The **<subject> verb infinitive** rule recognizes its pattern in the sentence and instantiates itself as **<subject> threatened to murder**. The resulting extraction pattern represents the expression "threatened to murder" and will extract the subject of the verb "threatened" as a perpetrator.

Text Id: DEV-MUC4-0071 **Targeted Noun Phrase:** "guerrillas"
Sentence: The Salvadoran guerrillas today threatened to murder individuals involved in 19 March presidential elections if they do not resign from their posts.

CONCEPT NODE
Name: perpetrator-subject-verb-infinitive-threatened-to-murder
Trigger: murder
Variable Slots: (perpetrator (*SUBJECT* 1))
Constraints: ((class PERPETRATOR *SUBJECT*)
Constant Slots: (type perpetrator)
Enabling Conditions: ((active) (trigger-preceded-by 'threatened 'to))

Fig. 5. Concept node for "<perpetrator> threatened to murder"

Although many of the concept nodes produced by AutoSlog represent relevant expressions, some of them represent expressions that are too general or do not make much sense (for example, this frequently happens when the sentence analyzer makes a mistake or when the pp-attachment algorithm chooses a bad attachment point). Therefore, we introduced a human in the loop to review all of the concept nodes created by AutoSlog. Using a simple interface, the user can quickly scan each of the extraction patterns (e.g., **"<perpetrator> threatened to murder"**) and determine whether it should be *accepted* for the final dictionary or *rejected* and thrown away.

We have applied AutoSlog to three domains: terrorism, joint ventures, and microelectronics. Since we did not have hand-crafted dictionaries for the joint ventures and microelectronics domains, it was difficult to evaluate the effectiveness of the AutoSlog dictionaries. The experiments described in the next section were done partly to address this issue. Our primary goal was to demonstrate that information extraction systems and resources are useful for natural language processing tasks besides straight information extraction. Our secondary goal was to show that the dictionaries created by AutoSlog are effective at making important domain discriminations. In the next section, we describe the text classification algorithm used in our experiments and present the results.

4 Text Classification Experiments

4.1 The Relevancy Signatures Algorithm

Text classification involves assigning category labels to texts. In these experiments, we focused on a binary classification problem: each text had to be labeled as either relevant or irrelevant to a given domain. The *relevancy signatures algorithm* [Riloff and Lehnert, 1994] was used for this task. Relevancy signatures were motivated by the observation that a single phrase is often enough to classify a text accurately. For example, the phrase "X was bombed" almost always describes a bombing, "X was kidnapped" almost always describes a kidnapping, and "assassination of X" almost always describes a murder. When processing texts in a limited domain, some expressions are reliable enough by themselves to warrant a relevant classification.

The relevancy signatures algorithm represents phrases as *signatures*, which are derived automatically from the concept nodes produced by CIRCUS. A signature is a word paired with a concept node. Together, this pair represents a unique set of linguistic expressions.[4] For example, the word "murdered" can be paired with the $murder-passive$ concept node to recognize passive forms of the verb "murdered", such as "X was murdered", "X and Y were murdered", and "X has been murdered." Similarly, the word "murdered" can be paired with the $murder-active$ concept node to recognize active forms of the verb "murdered", such as "X murdered Y" or "X has murdered Y."

The first step of the algorithm is to collect signatures by applying CIRCUS to a training corpus of preclassified texts. A set of *relevancy signatures* is then separated out using conditional probabilities. For each signature, we estimate the conditional probability that a text is relevant given that it generates the signature (that is, the number of occurrences of the signature in relevant texts divided by the total number of occurrences). Two thresholds are applied to identify the signatures that have the highest conditional probabilities: a relevancy threshold R and a frequency threshold M. A signature is deemed a relevancy signature is its conditional probability is $\geq$ R and its frequency is $\geq$ M. Intuitively, the relevancy signatures are a subset of signatures that were most highly correlated with relevant texts during training. Presumably, if a new text generates a relevancy signature then the text is likely to be relevant.

Although signatures represent only slightly more linguistic information than keywords, we have found that similar signatures can produce dramatically different classification results. Figure 6 shows several signatures, their estimated conditional probabilities (based on a training corpus of 1500 texts from the MUC-4 corpus [MUC-4 Proceedings, 1992]), and example sentences containing phrases represented by each signature. Figure 6 shows that 84% of texts containing the word "assassination" were relevant but only 49% of texts containing the word "assassinations" were relevant. In the MUC-4 terrorism domain, a text

[4] Pairing a word with a concept node is not necessary for AutoSlog's concept nodes because they already represent specific expressions. However, the hand-crafted dictionary contains more general concept nodes so this pairing is necessary.

is relevant only if it describes a *specific* terrorist incident. The singular noun, "assassination", usually refers to a specific assassination of a person or group of people, while the plural noun, "assassinations", often refers to assassinations in general (e.g., "The FMLN has claimed responsibility for many assassinations").

Signature	Prob.	Examples
<assassination, $murder$>	.84	the assassination of Hector Oqueli
<assassinations, $murder$>	.49	there were 50 assassinations in 1988
<bombed, $bombing-passive$>	.80	public buildings were bombed
<bombed, $bombing-active$>	.51	terrorists bombed two facilities
<casualties, $no-injury$>	.81	the attack resulted in no casualties
<casualties, $injury$>	.41	the officer reported 17 casualties
<dead, $found-dead-passive$>	1.00	the mayor was found dead
<dead, $left-dead$>	.61	the attack left 9 people dead
<dead, $number-dead$>	.47	the army sustained 9 dead

Fig. 6. Sample signatures and conditional probabilities

Figure 6 also shows that the passive form of "bombed" is more highly correlated with relevant texts than the active form. In the MUC-4 corpus, the active verb form is often used to describe military events but the passive verb form is more commonly used to describe terrorist events. One possible explanation for this phenomenon is that the perpetrator is often unknown in terrorist attacks. The passive verb form may also reflect a sense of victimization being conveyed by the reporters. One advantage of this approach is that these distinctions are identified automatically using statistics generated from a training corpus. It would be difficult, if not impossible, for a person to anticipate these differences.

To classify a new text, CIRCUS processes the text and the concept nodes instantiated during sentence processing are transformed into signatures. If any of the signatures is in the list of relevancy signatures for the domain, then the text is classified as relevant. If not, then the text is classified as irrelevant. We applied the relevancy signatures algorithm to three domains using concept node dictionaries created by AutoSlog. The results for these experiments are presented in the next three sections.

4.2 Results in the Terrorism Domain

Before testing AutoSlog's terrorism dictionary, we first trained the relevancy signatures algorithm using the hand-crafted MUC-3 concept node dictionary. The hand-crafted dictionary serves as a baseline for how well the relevancy signatures algorithm can perform using a manually encoded dictionary. 1500 texts (51% were relevant) from the MUC-4 corpus were used as training input for the relevancy signatures algorithm. After training was completed, testing was done on two blind sets of 100 texts each: TST3 and TST4. We ran the algorithm multiple times using a variety of different threshold settings: R was varied from

70 to 95 in increments of 5, and M was varied from 0 to 20 in increments of 1. Figure 7 shows the scatterplot generated from these runs.

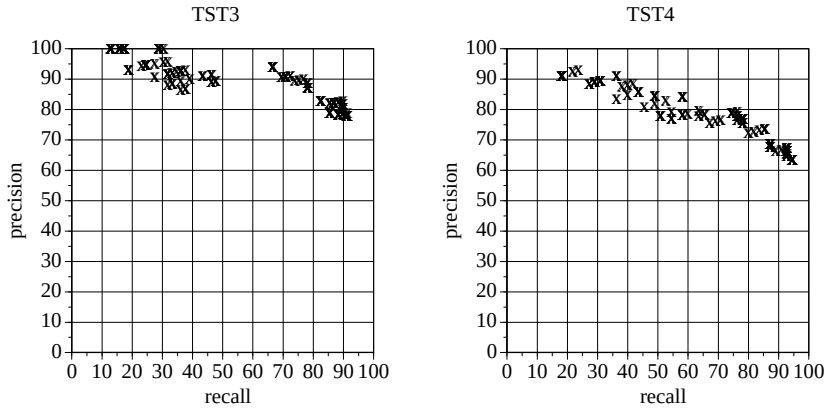

Fig. 7. Terrorism Results for the Hand-crafted Dictionary

Each data point represents a recall/precision pair for one set of threshold values. *Recall* is calculated as the number of texts correctly classified as relevant by the algorithm divided by the number of texts that should have been classified as relevant. *Precision* is defined as the number of texts correctly classified as relevant by the algorithm divided by the total number of texts classified as relevant by the algorithm. Recall and precision are metrics commonly used in the information retrieval community. There is almost always a tradeoff between recall and precision: achieving high recall usually means sacrificing precision and achieving high precision usually means sacrificing recall. Since there are some applications that demand high recall and others that demand high precision, we find it useful to show the spectrum of recall/precision levels that the algorithm can achieve.

Figure 7 shows that the relevancy signatures performed well in the terrorism domain using the hand-crafted dictionary. The relevancy signatures algorithm was designed with high-precision applications in mind, so the data points at the high precision end of the spectrum are the most interesting. If we look closely at a few of these data points, we see that the algorithm could achieve high precision with non-trivial levels of recall on both test sets. On TST3, the algorithm was able to achieve 100% precision with 30% recall and 94% precision with 67% recall. On TST4, the algorithm was able to obtain 93% precision with 24% recall, and 84% precision with 58% recall. In general, the relevancy signatures algorithm performed better on TST3 than TST4 but was able to achieve high precision on both test sets.

At the high recall end of the spectrum, we see one data point with 91% recall and 79% precision on TST3 and another data point with 94% recall and 63%

precision. These numbers should be interpreted with respect to the total number of relevant texts in each test set. TST3 contains 69 relevant texts and TST4 contains 55 relevant texts. So we could easily achieve 69% precision on TST3 and 55% precision on TST4 simply by classifying every text as relevant! Therefore it is worth noting that the relevancy signatures algorithm shows improvement over this baseline.

Next, we performed the same experiment but replaced the hand-crafted dictionary with the terrorism dictionary produced by AutoSlog. The MUC-4 corpus included 772 relevant terrorism texts and corresponding answer keys which were used as input to AutoSlog. AutoSlog produced 1237 unique concept node definitions, which were then filtered by a person. The person took about 5 hours to do the filtering and accepted 450 of the concept nodes for the final AutoSlog dictionary. The relevancy signatures algorithm was trained on the same set of 1500 texts.

Figure 8 shows the scatterplots for this experiment. There are two important observations to be made. (1) The scatterplots for the hand-crafted dictionary show more regular curves than the scatterplots for the AutoSlog dictionary. This is to be expected because the hand-crafted dictionary was built by a person who presumably created only relevant concept nodes. The AutoSlog dictionary, however, was constructed automatically and therefore may contain irrelevant concept nodes. Although the AutoSlog dictionary was manually filtered by a person, it is often difficult for a person to look quickly at an extraction pattern and accurately judge whether it will be useful for the domain. When a person goes to the trouble of defining a pattern by hand, however, presumably the person is motivated to do so because they have reason to believe that it is important.

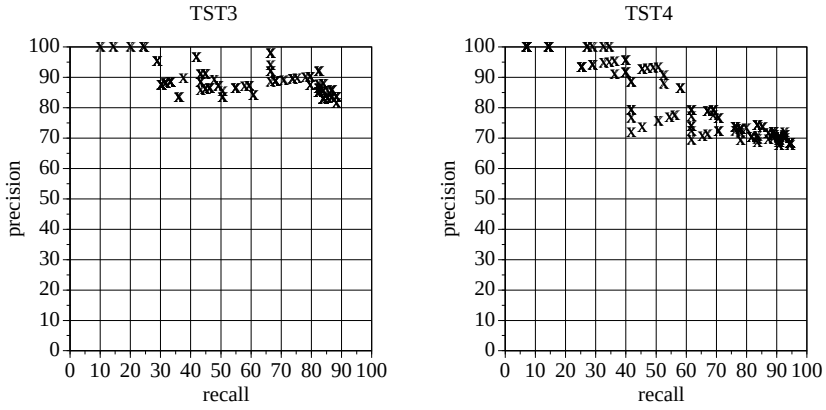

Fig. 8. Terrorism Results for the AutoSlog Dictionary

The second observation (2) is that the AutoSlog dictionary performed at least as well as the hand-crafted dictionary on TST3 and considerably better

on TST4. Many of the data points in Figure 7 are paralleled in Figure 8, and improved in several cases. The most notable improvement was on TST4. While the hand-crafted dictionary did not obtain 100% precision at any recall level, the AutoSlog dictionary produced several data points with 100% precision, including one with 35% recall. The AutoSlog dictionary also produced a data point with 93% precision and 51% recall. On TST3, the dictionaries showed similar performance. For example, the Autoslog dictionary also achieved 100% precision but with slightly lower recall of 25%. But it was able to achieve 98% precision with 67% recall which is better than the 94% precision with 67% recall produced by the hand-crafted dictionary.

Overall, the AutoSlog dictionary performed at least as well as the hand-crafted dictionary in the terrorism domain. This result suggests that the AutoSlog dictionary can duplicate most if not all of the functionality of the hand-crafted dictionary (with respect to the problem of text classification). And since the AutoSlog dictionary outperformed the hand-crafted dictionary in some cases, AutoSlog seems to have generated some extraction patterns that were useful for the domain but were not encoded in our hand-crafted dictionary.

4.3 Results in the Joint Ventures Domain

The joint ventures domain was based on the MUC-5 information extraction task, so we used the MUC-5 text corpora and answer keys for our experiments [MUC-5 Proceedings, 1993]. The MUC-5 joint ventures corpus contained 924 relevant texts plus corresponding answer keys, which were used as input to AutoSlog. AutoSlog generated 3167 concept nodes for the joint ventures domain; 944 of these were accepted for the final dictionary after manual filtering. We also automatically generated morphological variants for these concept nodes so the final joint ventures dictionary contained 2515 concept node definitions: the original 944 definitions plus morphological variants (see [Riloff, 1996] for details of this process).

For the text classification task, a text was generally considered to be relevant if it described a joint venture between two or more named entities (companies, governments, or people).[5] We used a corpus of 1200 preclassified texts (54% relevant[6]) to train the relevancy signatures algorithm. However, we did not have separate blind test sets for this domain, so we used a 10-fold cross validation design to evaluate the performance of the algorithm. We also used an empirical method analogous to cross-validation to determine the best threshold values for 7 different recall/precision settings. (The details of this procedure are beyond the scope of this paper but have been discussed elsewhere [Riloff and Lehnert, 1994].) Figure 9 shows the result of this experiment. Each data point represents one of the 7 recall/precision results achieved by the algorithm for the empirically derived threshold values.

[5] The official guidelines for relevance are explained in the MUC-5 proceedings [MUC-5 Proceedings, 1993].

[6] Most of the irrelevant texts were drawn from the Tipster detection corpus [Tipster Proceedings, 1993].

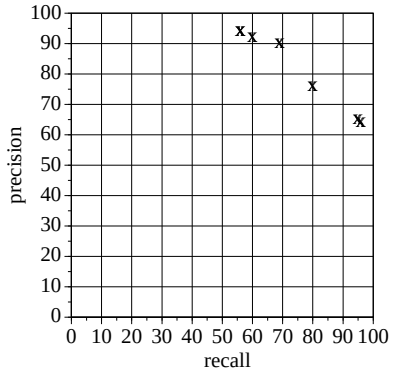

Fig. 9. Joint Venture Results

Figure 9 shows that the relevancy signatures algorithm performed very well in the joint ventures domain using the AutoSlog dictionary. At the high precision end of the spectrum, we see one data point that represents 94% precision with 56% recall and another that represents 92% precision with 60% recall. It is important to keep in mind that these results are based on 1200 texts (using the 10-fold cross-validation design), whereas the terrorism experiments were based on test sets containing a total of 200 texts. Therefore, these results provide strong evidence that AutoSlog produced an effective joint ventures dictionary for the text classification task.

4.4 Results in the Microelectronics Domain

The third text classification experiment involved the MUC-5 microelectronics domain. The MUC-5 microelectronics corpus contained 787 relevant microelectronics texts plus corresponding answer keys that were used as input to AutoSlog. Using this corpus, AutoSlog generated 2952 concept nodes for the microelectronics domain. We manually filtered only some of the concept nodes produced by AutoSlog for this domain (see [Riloff, 1996] for details about this process). After selective filtering and generating morphological variants (as per the joint ventures domain), the final microelectronics dictionary contained 4220 concept node definitions.

In the MUC-5 microelectronics domain, a text was generally defined to be relevant if it mentioned a microelectronics process linked to a specific company or research group. We used 500 texts from the MUC-5 corpus (57% relevant) to train the relevancy signatures algorithm. We then used the same cross-validation design used for the joint ventures domain to evaluate the performance of the dictionary.

Figure 10 shows the results of the microelectronics experiment. The text classification results for this domain were considerably weaker than they were for the other two domains. However, the AutoSlog microelectronics dictionary did

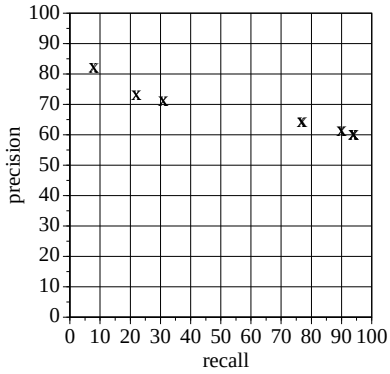

Fig. 10. Microelectronics Results

achieve respectable performance. In particular, we see one data point that represents 82% precision with 8% recall and another data point that represents 73% precision with 22% recall. Since the training corpus contained only 57% relevant texts, these precision results represent a non-trivial improvement over the baseline. Furthermore, the text classification training corpus for the microelectronics domain was much smaller than the corpora used for the terrorism and joint ventures domains (500 microelectronics texts, compared with 1500 terrorism texts and 1200 joint venture texts). Since the relevancy signatures algorithm is based on probability estimates, we expect its performance to be highly dependent on the size of the training corpus.

5 Related Work and Discussion

AutoSlog learns concept node definitions automatically using an annotated training corpus. However, it does not fall neatly into any of the common machine learning pigeonholes. AutoSlog is a one-shot learning system because it generates a complete extraction pattern from a single training instance. Therefore, in one sense, the closest points of comparison in the machine learning community are explanation-based learning (EBL) systems [DeJong and Mooney, 1986; Mitchell *et al.*, 1986] since EBL systems are able to produce complete concept representations from a single training instance. This is in contrast to inductive learning techniques that incrementally build a concept representation in response to multiple training instances (e.g., [Fisher, 1987; Quinlan, 1986; Utgoff, 1988]). AutoSlog does not rely on an explicit domain theory like most EBL systems, but it does use a set of domain-independent rules based on general syntactic properties of natural language. One important feature of AutoSlog is that its input (text and tagged noun phrases) is of a completely different form than its output (extraction patterns).

The one-shot learning aspect of AutoSlog distinguishes it from most other lexical acquisition systems, which build new definitions based on known definitions

of other words in the sentence or surrounding context (e.g., [Carbonell, 1979; Granger, 1977; Jacobs and Zernik, 1988]). In contrast, AutoSlog builds new dictionary definitions completely from scratch and depends only on a part-of-speech lexicon, which can be readily obtained from machine-readable dictionaries. AutoSlog is closely related to the PALKA system [Kim and Moldovan, 1993], which also learns structures for information extraction. However, PALKA uses a set of predefined keywords and frame definitions for the domain as a starting point and depends on semantic features associated with words for learning. The CRYSTAL system ([Soderland *et al.*, 1995] and Soderland et al., this volume) and LIEP (Huffman, this volume) also rely on semantic features to learn extraction patterns. AutoSlog discovers the trigger words for case frames on its own and does not require semantic features. Furthermore, a recent extension of AutoSlog, called AutoSlog-TS [Riloff and Shoen, 1995], eliminates the need for detailed text annotations from a user and requires only a training corpus of preclassified texts.

AutoSlog's ability to learn dictionaries of extraction patterns can substantially reduce the time required to build an information extraction system for a new domain. We have also demonstrated that dictionaries created by AutoSlog can be useful for other natural language processing tasks as well. These results suggest that the patterns learned by AutoSlog represent important domain concepts that may be applicable to many different problems, and that information extraction technology may be useful for a variety of language processing tasks.

Acknowledgements

This research was funded by NSF Grant no. EEC-9209623, supporting the Center for Intelligent Information Retrieval at the University of Massachusetts, NSF grant MIP-9023174, and NSF grant IRI-9509820.

References

Carbonell, J. G. 1979. Towards a Self-Extending Parser. In *Proceedings of the 17th Meeting of the Association for Computational Linguistics.* 3–7.

DeJong, Gerald and Mooney, R. 1986. Explanation-Based Learning: An Alternative View. *Machine Learning* 1:145–176.

Fisher, D. H. 1987. Knowledge Acquisition Via Incremental Conceptual Clustering. *Machine Learning* 2:139–172.

Granger, R. H. 1977. FOUL-UP: A Program that Figures Out Meanings of Words from Context. In *Proceedings of the Fifth International Joint Conference on Artificial Intelligence.* 172–178.

Jacobs, Paul and Rau, Lisa 1990. SCISOR: Extracting Information from On-Line News. *Communications of the ACM* 33(11):88–97.

Jacobs, P. and Zernik, U. 1988. Acquiring Lexical Knowledge from Text: A Case Study. In *Proceedings of the Seventh National Conference on Artificial Intelligence.* 739–744.

Kim, J. and Moldovan, D. 1993. Acquisition of Semantic Patterns for Information Extraction from Corpora. In *Proceedings of the Ninth IEEE Conference on Artificial*

Intelligence for Applications, Los Alamitos, CA. IEEE Computer Society Press. 171–176.

Lehnert, W. G. and Sundheim, B. 1991. A Performance Evaluation of Text Analysis Technologies. *AI Magazine* 12(3):81–94.

Lehnert, W.; Cardie, C.; Fisher, D.; Riloff, E.; and Williams, R. 1991. University of Massachusetts: Description of the CIRCUS System as Used for MUC-3. In *Proceedings of the Third Message Understanding Conference (MUC-3)*, San Mateo, CA. Morgan Kaufmann. 223–233.

Lehnert, W.; Cardie, C.; Fisher, D.; McCarthy, J.; Riloff, E.; and Soderland, S. 1992. University of Massachusetts: MUC-4 Test Results and Analysis. In *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, San Mateo, CA. Morgan Kaufmann. 151–158.

Lehnert, W. 1991. Symbolic/Subsymbolic Sentence Analysis: Exploiting the Best of Two Worlds. In Barnden, J. and Pollack, J., editors 1991, *Advances in Connectionist and Neural Computation Theory, Vol. 1.* Ablex Publishers, Norwood, NJ. 135–164.

Mitchell, T. M.; Keller, R.; and Kedar-Cabelli, S. 1986. Explanation-Based Generalization: A Unifying View. *Machine Learning* 1:47–80.

Proceedings of the Third Message Understanding Conference (MUC-3), San Mateo, CA. Morgan Kaufmann.

Proceedings of the Fourth Message Understanding Conference (MUC-4), San Mateo, CA. Morgan Kaufmann.

Proceedings of the Fifth Message Understanding Conference (MUC-5), San Francisco, CA. Morgan Kaufmann.

Quinlan, J. R. 1986. Induction of Decision Trees. *Machine Learning* 1:80–106.

Riloff, E. and Lehnert, W. 1994. Information Extraction as a Basis for High-Precision Text Classification. *ACM Transactions on Information Systems* 12(3):296–333.

Riloff, E. and Shoen, J. 1995. Automatically Acquiring Conceptual Patterns Without an Annotated Corpus. In *Proceedings of the Third Workshop on Very Large Corpora.* 148–161.

Riloff, E. 1993. Automatically Constructing a Dictionary for Information Extraction Tasks. In *Proceedings of the Eleventh National Conference on Artificial Intelligence.* AAAI Press/The MIT Press. 811–816.

Riloff, E. 1996. An Empirical Study of Automated Dictionary Construction for Information Extraction in Three Domains. *Artificial Intelligence.* To appear.

Soderland, S.; Fisher, D.; Aseltine, J.; and Lehnert, W. 1995. CRYSTAL: Inducing a conceptual dictionary. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence.* 1314–1319.

Proceedings of the TIPSTER Text Program (Phase I), San Francisco, CA. Morgan Kaufmann.

Utgoff, P. 1988. ID5: An Incremental ID3. In *Proceedings of the Fifth International Conference on Machine Learning.* 107–120.