In Natural Language Information Retrieval, forthcoming from Kluwer Academic Publishers.

EXTRACTION-BASED TEXT CATEGORIZATION: GENERATING DOMAIN-SPECIFIC ROLE RELATIONSHIPS AUTOMATICALLY

ELLEN RILOFF AND JEFFREY LORENZEN
Department of Computer Science
University of Utah
Salt Lake City, UT 84112
{*riloff,jlorenz*}*@cs.utah.edu*

Abstract. In previous work, we developed several algorithms that use information extraction techniques to achieve high-precision text categorization. The *relevancy signatures* algorithm classifies texts using extraction patterns, and the *augmented relevancy signatures* algorithm classifies texts using extraction patterns and semantic features associated with role fillers (Riloff and Lehnert, 1994). These algorithms relied on hand-coded training data, including annotated texts and a semantic dictionary. In this chapter, we describe two advances that significantly improve the practicality of our approach. First, we explain how the extraction patterns can be generated automatically using only preclassified texts as input. Second, we present the *word-augmented relevancy signatures* algorithm that uses lexical items to represent domain-specific role relationships instead of semantic features. Using these techniques, we can automatically build text categorization systems that benefit from domain-specific natural language processing.

Natural language processing and information retrieval evolved as separate research areas. To a large extent, the difference between them revolves around the amount of knowledge that is used. Natural language processing (NLP) techniques rely on syntactic and semantic knowledge that is often manually encoded for a particular topic. Information retrieval (IR) techniques typically use general methods for processing large volumes of text based on statistical word models. These two paradigms reflect a tug-of-war

between the benefits of domain knowledge and the costs and constraints associated with it.

Our goal has been to bridge this gap by developing methods to acquire domain knowledge for NLP automatically. Our work focuses on a natural language processing task called *information extraction* (IE), which aims to extract domain-specific information from text. Recent advances in this area have made it possible to build IE systems with minimal knowledge engineering. For the last few years, we have been using information extraction techniques to build text categorization systems that benefit from natural language processing while requiring little or no manual knowledge engineering.

In previous work, we demonstrated that information extraction techniques could be used to achieve high-precision text categorization in some domains. We used *relevancy signatures* to classify texts on the basis of linguistically-motivated extraction patterns, and *augmented relevancy signatures* to classify texts using extraction patterns and semantic features associated with role fillers (Riloff and Lehnert, 1994). Both of these algorithms performed better than a comparable word-based algorithm in terrorism and joint venture domains (Riloff, 1994). Much of the domain knowledge needed for these algorithms could be acquired automatically, but there were still significant knowledge-engineering bottlenecks lurking underneath. A manually annotated training corpus was required to generate the extraction patterns, and a dictionary of words tagged with semantic features was needed for the augmented relevancy signatures.

In this article, we describe two advances that significantly improve the practicality of our approach. First, we explain how the extraction patterns can be generated automatically using a system called AutoSlog-TS, without the need for an annotated training corpus. Second, we present a variation of the augmented relevancy signatures algorithm that does not rely on semantic features. The new version uses only lexical items to represent role fillers, which are extracted automatically from the training corpus. Together with AutoSlog-TS, the *word-augmented relevancy signatures* represent domain-specific role relationships that are generated automatically without any manual knowledge engineering.

In the first section, we describe the extraction patterns used by our natural language processing system and describe two previous text representations based on these extraction patterns, *relevancy signatures* and *augmented relevancy signatures*. In the second section, we discuss our dictionary construction system, AutoSlog-TS, that can generate extraction patterns automatically. The only input to the system is a set of "preclassified" texts associated with the domain. In the third section, we present the new word-augmented relevancy signatures representation and algorithm.

Finally, we present experiments comparing the different text representations on four categorization tasks related to terrorism. In all cases, the signature representations performed better than individual words and the word-augmented signatures performed comparably to the augmented relevancy signatures with semantic features.

1. Extraction-based text categorization

1.1. EXTRACTION PATTERNS

Information extraction (IE) is a natural language processing task that has received growing attention in recent years, largely due to the government-sponsored Message Understanding Conferences (MUCs) (MUC-5 Proceedings, 1993; MUC-4 Proceedings, 1992; MUC-3 Proceedings, 1991). IE systems extract domain-specific information from natural language text. The domain and types of information to be extracted must be defined in advance. IE systems often focus on object identification, such as references to people, places, companies, and physical objects. For example, the MUC-4 terrorism domain involved the extraction of perpetrators, victims, physical targets, weapons, dates, and locations (MUC-4 Proceedings, 1992). The MUC-5 joint ventures domain required the extraction of companies, people, monetary values, and facilities associated with joint ventures (MUC-5 Proceedings, 1993).

Most IE systems use some form of partial parsing to recognize local syntactic constructs without generating a complete parse tree for each sentence. Since IE systems are usually applied to large sets of real documents, the robustness provided by partial parsing is often essential to cope with unknown words and ungrammatical or ill-formed text. Domain-specific *extraction patterns* (or something similar) are used to identify relevant information. An important aspect of the information extraction paradigm is that irrelevant information is ignored. If no extraction patterns apply to a sentence, then nothing will be extracted. Consequently, irrelevant paragraphs and documents can be processed very quickly.[1]

The experiments presented in this article were done using a conceptual sentence analyzer called CIRCUS (Lehnert, 1991). CIRCUS uses case frames to extract information, but we will refer to these case frames as extraction patterns for our purposes (it should be noted, however, that CIRCUS' case frames do more than we describe here). Each extraction pattern is indexed by a *trigger word* and contains a set of activating conditions and case roles (slots). The trigger word is a lexical item that indexes the

[1]The extraction patterns can be indexed under the words that trigger them, so any sentence not containing one of these trigger words does not have to be processed.

pattern, and the activating conditions are additional constraints that must be satisfied for the pattern to be applied. Intuitively, each extraction pattern represents linguistic expressions that are defined by the trigger word and activating conditions together. For example, consider the following sentence:

FMLN terrorists in retaliation for recent arrests attempted to kill 5 policemen with car bombs.

In a terrorism domain, we might want to extract three items from this sentence: the perpetrators (*FMLN terrorists*), the victims (*5 policemen*), and the weapons (*car bombs*). The following extraction patterns would do the job:

<perpetrator> attempted to kill
attempted to kill <victim>
to kill with <weapon>

The first extraction pattern is triggered by the word *kill* and its activating conditions check that *kill* is in an infinitive form and is preceded by the verb *attempted*. If these conditions are satisfied, then the subject of the verb *attempted* is extracted as a perpetrator. Note that the pattern extracts *FMLN terrorists* because it is the subject of the verb *attempted*, even though the phrase *in retaliation for recent arrests* physically separates them. Although we refer to the case frames as patterns, it is important to remember that extracting information depends on syntactic processing. We are not simply extracting adjacent words!

The second extraction pattern has the same trigger word *kill* and activating conditions, but it extracts the object of *kill* as a victim. Both patterns represent the expression *attempted to kill* and could be merged into a single case frame with two slots, perpetrator and victim. For now, we will assume that each pattern has just a single slot to extract one piece of information.

The third extraction pattern is also triggered by the word *kill* but its activating conditions check that *kill* is in an infinitive form and is followed by the preposition *with*. The object of the preposition is extracted as a weapon. Note that *with* does not have to be adjacent to *kill*, it simply has to follow it in the same clause. Once again, this example illustrates the natural language processing capabilities. The extraction pattern is essentially telling the system to attach a prepositional phrase containing the preposition *with* to the verb *kill*. There may be many words in between the prepositional phrase and its attachment point; we are not simply searching for adjacent words. The types of extraction patterns currently recognized by our system

are shown in Figure 1. An example instantiation of each pattern is shown on the right, with its trigger word in italics.

EXTRACTION PATTERN	**EXAMPLE**
<subj> passive-verb	<victim> was *murdered*
<subj> active-verb	<perpetrator> *bombed*
<subj> active-verb dobj	<perpetrator> *threw* dynamite
<subj> verb infinitive	<perpetrator> attempted to *kill*
<subj> aux noun	<victim> was *victim*
active-verb <dobj>	*bombed* <target>
infinitive <dobj>	to *kill* <victim>
verb infinitive <dobj>	tried to *attack* <target>
gerund <dobj>	*killing* <victim>
noun aux <dobj>	*fatality* was <victim>
noun prep <np>	*bomb* against <target>
active-verb prep <np>	*killed* with <weapon>
passive-verb prep <np>	was *aimed* at <target>
infinitive prep <np>	to *assassinate* with <weapon>

Figure 1. Extraction pattern types

In most IE systems, extraction patterns for the domain of interest are defined by hand. This is a tedious and time-consuming process. For the MUC-4 terrorism domain, we estimated that it took approximately 1500 person-hours to define the extraction patterns. Fortunately, several systems have been developed recently that can generate extraction patterns automatically or semi-automatically from training data (Kim and Moldovan, 1993; Soderland *et al.*, 1995; Huffman, 1996), including our own AutoSlog system (Riloff, 1996a). AutoSlog constructed a dictionary of extraction patterns for the MUC-4 terrorism domain that achieved 98% of the performance of hand-crafted patterns. AutoSlog uses annotated texts for training and a human must manually review the resulting extraction patterns. The manual review process is typically fast (it took ~5 hours to review over 1200 terrorism patterns), but the need for annotated training texts is a serious knowledge-engineering bottleneck. In Section 2, we describe a new version of AutoSlog, called AutoSlog-TS, that generates domain-specific extraction patterns using relevant and irrelevant sample texts without annotations. AutoSlog-TS is the first system that can generate extraction patterns using only raw text as input.

1.2. RELEVANCY SIGNATURES

Extraction patterns represent local linguistic expressions that are slightly more sophisticated than keywords. Extraction patterns depend on syntactic contexts surrounding a word. For example, several extraction patterns might be triggered by the word *bomb* to recognize its active verb form (*will bomb*), its infinitive verb form (*to bomb*), or the noun form (*a bomb*). Prepositions are also important for many patterns because they often identify role relationships. For example, *killed by X* suggests that X is a perpetrator, while *killed with X* suggests that X is a weapon. We have shown elsewhere that recognizing different verb forms and prepositions can dramatically improve the accuracy of some classification terms (Riloff, 1995).

The linguistic expressions recognized by an extraction pattern can be represented as a *signature*. A signature is simply an extraction pattern paired with its trigger word. For example, a signature that pairs passive-verb activating conditions with the trigger *bombed* would represent the phrases *was bombed, were bombed,* and *has been bombed.* Similarly, a signature that pairs active-verb activating conditions with the trigger word *bombed* would represent expressions such as *has bombed* or *have bombed.* A signature is a concise representation of the linguistic expressions recognized by an instantiated extraction pattern.

Most signatures, however, are not useful for classification. For example, in the terrorism domain we need signatures that refer to death because people are often killed during terrorist attacks. But people are also killed in many other ways, such as accidents, military incidents, and random violence. Consequently, signatures representing the expressions *was killed* or *has died* are not particularly helpful for identifying terrorism texts.

To find useful classification terms, we need to identify signatures that are representative of the domain. We will refer to these signatures as *relevancy signatures*. Intuitively, relevancy signatures represent expressions that are much more common in one domain than in others. We use a conditional probability estimate to find the relevancy signatures for a domain. A training corpus consisting of relevant and irrelevant sample texts is used to generate the probability estimates. The procedure is:

1. Run the sentence analyzer over the training corpus using extraction patterns developed for the domain. Keep track of the signatures generated by each text.
2. For each signature, compute the conditional probability that a text is relevant given that it contains that signature. This is simply the number of times the signature appeared in relevant texts divided by the total number of times the signature appeared in the training corpus. For clarity, we will refer to this conditional probability as the *relevance*

rate of a signature.

3. Select a set of relevancy signatures by specifying two thresholds: a minimum acceptable relevance rate (R), and a minimum acceptable frequency value (F). For example, R=.80 and F=10 means that a signature becomes a relevancy signature only if it occurred at least 10 times in the training corpus and at least 80% of those occurrences appeared in relevant texts.

Once a set of relevancy signatures is identified for a domain, the classification procedure is simple. Given new texts to classify, we run the sentence analyzer over the texts using the same extraction patterns and we record the signatures generated by each text. If any relevancy signatures are found in a text, then that text is classified as relevant. If no relevancy signatures are found, then the text is classified as irrelevant.

Note that it takes only a single relevancy signature to classify a text as relevant. And a text with only one relevancy signature is considered to be just as relevant as a text with 20 relevancy signatures. This approach is different from most information retrieval algorithms that rank texts based on multiple indexing terms (e.g., (Turtle and Croft, 1991; Salton, 1971)). There are two reasons for our approach. First, the MUC-4 terrorism domain is defined such that a text with even a passing reference to a terrorist incident is relevant.[2] The MUC-4 corpus contains many texts that are relevant because a single sentence refers to a terrorist attack. Humans have little trouble classifying these texts as relevant because there is usually an obvious reference to a terrorist attack, so we reasoned that our system should be able to do the same.

The second justification for our approach is that we are doing text categorization, not query-based information retrieval. For text categorization the texts do not need to be ranked, just classified as relevant or irrelevant. But every text must be accurately classified to produce good results. For example, suppose a large text collection contains 100 relevant texts and an IR system ranks 50 of them near the top and 50 near the bottom. In a query-based scenario, the user quickly sees 50 relevant documents so the system would probably be perceived as working well. But in a text classification scenario, the system would have misclassified 50% of the relevant documents. Both the domain and the task play a significant role in deciding what approach is best. We performed a few experiments to classify texts based on multiple signatures and did not get consistently better results than with the single signature approach (Riloff, 1994).

Another reason why the single signature approach is effective is that the signatures capture some linguistic context. To illustrate the power of

[2]The notion of what constitutes terrorism is a separate issue that is explained in the MUC-4 guidelines (MUC-4 Proceedings, 1992).

signatures over keywords, suppose we set R=.90 and F=20. We find 12 relevancy signatures for the terrorism domain, generated from a training corpus of 1500 MUC-4 texts. Figure 2 lists the 12 terrorism relevancy signatures and their statistics. Intuitively, the signatures represent expressions associated with terrorism, but it is logical to wonder whether the trigger words alone would be just as effective. On the righthand side of Figure 2, we show the trigger words and their respective statistics, generated from the same training corpus but without considering the surrounding linguistic context.

Signature	**Rel**	**Freq**	**Trigger**	**Rel**	**Freq**
<subj> was shot_to_death	.96	28	shot_to_death	.97	31
<subj> went_off	.96	25	went_off	.96	26
murder on <pp>	.95	20	murder	.76	308
was murdered on <pp>	.93	29	murdered	.76	204
was kidnapped in <pp>	.92	26	kidnapped	.83	151
located on <pp>	.92	25	located	.77	180
blew_up <dobj>	.92	25	blew_up	.89	28
to protest <dobj>	.92	24	protest	.66	62
was found on <pp>	.91	23	found	.62	158
exploded at <pp>	.90	21	exploded	.90	118
bomb on <pp>	.90	21	bomb	.84	204
<subj> exploded	.90	113	exploded	.90	118

Figure 2. Relevancy signatures and trigger word statistics for terrorism

Some of the triggers, such as shot_to_death and went_off, are very effective by themselves.[3] However, most of the trigger words have substantially lower relevance rates than their respective signatures. In particular, signatures that contain the preposition *on* generally have a much higher correlation with relevant texts than their corresponding trigger words. We believe that this is because objects of the preposition *on* are often date expressions. For example, *murder on X* usually refers to a murder on a specific date. The preposition helps to distinguish specific murder incidents from general references to murder (e.g., *murder is increasing across the nation*).

[3]The underscored words represent phrases that are treated as lexical items by CIRCUS. These phrases were part of a phrasal lexicon that was hand-crafted for the MUC-4 terrorism domain. We left them in our system because the hand-crafted extraction patterns need them and removing them would have put the automatically generated extraction patterns at a disadvantage (we compare these patterns in Section 4.5). In retrospect, we expect that AutoSlog-TS could have generated extraction patterns to recognize most of these expressions on its own.

Also, the infinitive form of *protest* is much more strongly correlated with terrorism than the word *protest* alone. We have found many examples, in multiple domains, where different verb forms or different prepositions following a verb can dramatically change the effectiveness of classification terms (Riloff, 1995).

1.3. AUGMENTED RELEVANCY SIGNATURES

Signatures can be more powerful than individual words because they represent local syntactic context, but a major limitation is that they do not include role fillers. A signature may represent a verb's expectation for a prepositional phrase, but it does not represent the object of the prepositional phrase. In some domains, role fillers are critical. For example, the MUC-4 terrorism domain is defined such that a victim of a terrorist act must be civilian. If a terrorist group attacks military personnel or other terrorists, then the act is not classified as terrorism. The sentence *terrorists killed shoppers* would be relevant but the sentence *terrorists killed soldiers* would not. Although this definition is arbitrary, it is easy to imagine many domain descriptions that depend on certain types of role fillers.

Intuitively, we want to classify a text as relevant if it contains both a relevant key phrase and a relevant role filler. The *augmented relevancy signatures* algorithm (Riloff and Lehnert, 1994) does this by using both relevancy signatures and *relevant slot triples*. The triples are of the form (event-type slot-type feature) and are generated automatically from each extracted role filler. The feature corresponds to the semantic feature associated with the head noun of the role filler. For example, suppose the word *terrorists* was tagged with a TERRORIST feature in the dictionary and the word *soldiers* was tagged with a MILITARY feature. The sentence *"Armed terrorists killed 50 soldiers"* would yield two signatures representing the expressions "<subject> killed" and "killed <dobj>" and two slot triples representing the terrorists (murder, perpetrator, TERRORIST) and the soldiers (murder, victim, MILITARY).

The *augmented relevancy signatures* algorithm is analogous to the relevancy signatures algorithm except that we operate on both signatures and slot triples. During training, we collect all the signatures and slot triples generated by the extraction patterns for each text. We compute the relevance rate (conditional probability) and overall frequency of each slot triple in addition to each signature. We then choose two sets of R and F thresholds, one for the signatures and one for the triples, to identify a set of relevancy signatures and a set of relevant slot triples. Figure 3 shows the relevant slot triples that were identified using the thresholds R=.85, F=50.

The slot triples in Figure 3 illustrate some of the semantic properties

of the domain. As we would expect, we see that bombing and kidnapping events with terrorist perpetrators are usually relevant. We also get a sense of the common victims of terrorism. Murders of politicians, judges, government officials, and clergy were highly correlated with terrorism.[4] And we see that bombs and dynamite are common weapons in terrorist attacks.

Slot Triple	**Rel**	**Freq**
(bombing instrument BOMB)	.93	147
(bombing instrument DYNAMITE)	.92	52
(bombing perpetrator TERRORIST)	.92	52
(kidnapping perpetrator TERRORIST)	.92	52
(murder victim POLITICIAN)	.89	103
(weapon[5] instrument DYNAMITE)	.89	111
(kidnapping victim PROPER-NAME)	.89	90
(murder victim LEGAL-OR-JUDICIAL)	.88	51
(murder victim GOVT-OFFICIAL)	.88	120
(murder victim CLERGY)	.85	266
(kidnapping victim CIVILIAN)	.85	54

Figure 3. Relevant slot triples for terrorism

To classify a new text, we run the text through the sentence analyzer and keep track of the signatures and slot triples that are produced. Each signature has an associated role filler representing the information that was extracted by its pattern, so we pair each signature with the slot triple corresponding to its role filler. For example, the sentence *"terrorists killed soldiers"* would produce the signature "<subject> killed" paired with the slot triple (murder, perpetrator, TERRORIST) and the signature "killed <dobj>" paired with the triple (murder, victim, MILITARY). This pairing is important because the signature and slot triple together function as a complex classification term representing a key phrase and the semantics associated with its role filler.

For each signature/slot-triple pair, we check to see whether the signature is in the list of relevancy signatures and the slot triple is in the list of relevant triples. If both of these conditions are satisfied, then the text is classified as relevant. If not, then the text is classified as irrelevant. This algorithm

[4]The clergy victims refer to several Jesuit clergy who were murdered by terrorists in El Salvador.

[5]The hand-crafted terrorism dictionary contained several patterns to extract weapons, which were labeled simply as "weapon" patterns that were activated in many event contexts.

is very conservative about classifying a text as relevant: a text must have both a key phrase and an associated role filler that were strongly correlated with relevant texts during training.

Augmented relevancy signatures performed well in previous experiments (Riloff and Lehnert, 1994), but they depend on both extraction patterns for the domain and a dictionary of words tagged with semantic features. In the following sections, we present new methods for eliminating these knowledge-engineering bottlenecks. In Section 2, we describe a system called AutoSlog-TS that generates extraction patterns for a domain using only preclassified texts. And in Section 3 we present the *word-augmented relevancy signatures* algorithm that uses lexical items to capture role relationships instead of semantic features.

2. Automatically generating extraction patterns

As we mentioned in Section 1.1, AutoSlog was a major step forward in automating the construction of extraction patterns. But AutoSlog needs specialized training data: answer keys or annotated texts in which the noun phrases that should be extracted have been labeled. For example, in the terrorism domain, noun phrases that refer to perpetrators, targets, victims, and weapons were marked. Generating an annotated training corpus is both time-consuming and deceptively difficult (determining which words to mark is tricky), and a new training corpus must be annotated for each domain.

To further reduce this knowledge-engineering bottleneck, we have developed a successor to AutoSlog, called AutoSlog-TS (Riloff, 1996b), that creates extraction patterns from raw text. As input, AutoSlog-TS needs *preclassified* texts that have been identified as relevant or irrelevant to the domain. Nothing inside the texts needs to be marked in any way. AutoSlog-TS is illustrated in Figure 4.[6]

In Stage 1, AutoSlog-TS uses a slightly modified version of AutoSlog's heuristic rules to generate an extraction pattern for every noun phrase in the corpus.[7] At the end of Stage 1, we have a giant dictionary of extraction patterns that are literally capable of extracting every noun phrase in the corpus. In Stage 2, we process the training corpus a second time using the new extraction patterns. For each pattern, we estimate the conditional probability that a text is relevant given that it activates the pattern. We call this value an extraction pattern's *relevance rate*. Then we rank them by

[6] The "concept nodes" in Figure 4 refer to CIRCUS' case frames (i.e., the extraction patterns).

[7] In contrast to AutoSlog, multiple rules can fire in response to the same noun phrase, creating several patterns to extract the same item. The statistics will later reveal which pattern is statistically preferable.

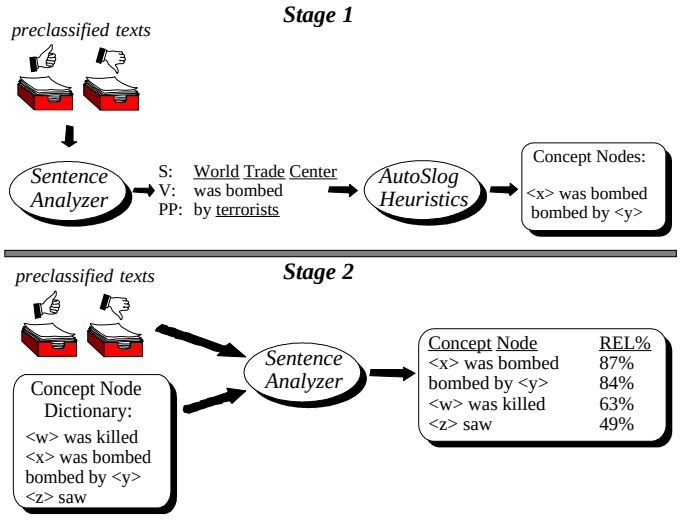


Figure 4. AutoSlog-TS flowchart

assigning each pattern a score corresponding to the product of its relevance rate and the log_2 of its total frequency.

In experiments with the MUC-4 terrorism corpus, AutoSlog-TS generated 32,345 unique extraction patterns from 1500 texts. In Stage 2, we discarded all patterns with frequency = 1 to keep the size of the dictionary manageable, reprocessed the training corpus with the remaining 11,225 patterns, computed their relevance rates, and ranked them. The expressions representing the top 25 extraction patterns appear in Figure 5. Most of these patterns are clearly associated with terrorism, so the ranking function did a good job of promoting the domain-specific expressions to the top.

For information extraction, we need to manually review and label the top patterns to get a reliable set of extraction patterns. Each pattern must be assigned an event type and a slot type. For example, the extraction pattern "<subject> was murdered" would be labeled as a murder event with a victim slot because it will extract murder victims. For text categorization, however, we hypothesized that we did not need to manually review the patterns. Our assumption was that we do not need to know what types of objects are extracted; we only care whether the extracted information is strongly associated with the domain. So we gave all of the extraction patterns to the classification algorithms and let them decide which patterns to use for categorization. All 11,225 case frames generated by AutoSlog-TS with frequency > 1 were given to the text categorization algorithms for training.

1. <subj> exploded	14. <subj> occurred
2. murder of <np>	15. <subj> was located
3. assassination of <np>	16. took_place on <np>
4. <subj> was killed	17. responsibility for <np>
5. <subj> was kidnapped	18. occurred on <np>
6. attack on <np>	19. was wounded in <np>
7. <subj> was injured	20. destroyed <dobj>
8. exploded in <np>	21. <subj> was murdered
9. death of <np>	22. one of <np>
10. <subj> took_place	23. <subj> kidnapped
11. caused <dobj>	24. exploded on <np>
12. claimed <dobj>	25. <subj> died
13. <subj> was wounded	

Figure 5. The top 25 extraction patterns

AutoSlog-TS is the first system that can generate extraction patterns using only raw text as input. AutoSlog-TS needs both relevant and irrelevant sample texts to decide which patterns are most strongly associated with the domain. Not coincidentally, the preclassified corpus needed for AutoSlog-TS is exactly the same input that is required for the text categorization algorithms. We exploit the preclassified texts by processing them twice: once to generate extraction patterns and once to apply the extraction patterns to the texts. The extracted information, in the form of signatures and role fillers, is then analyzed statistically to identify classification terms that are highly correlated with a category.

3. Word-augmented relevancy signatures

Augmenting relevancy signatures with semantic features produced much better results than relevancy signatures alone in the MUC-4 terrorism domain (Riloff and Lehnert, 1994). But there was a price to pay. Augmented relevancy signatures need a semantic feature hierarchy and a dictionary of words tagged with semantic features. Consequently, using augmented relevancy signatures in a new domain requires an initial time investment that might not be acceptable for many applications. To eliminate the need for semantic features, we investigated whether the role fillers could be represented using lexical items instead of semantic features. For clarity, we will refer to this new representation as *word-augmented signatures* and the original representation as *feature-augmented signatures.*

Feature-augmented signatures consist of signatures paired with slot

triples. Each slot triple represents the event-type and slot-type of the extraction pattern and a semantic feature representing its role filler. For information extraction, it is essential to understand what is being extracted. For text categorization, it was not clear whether this information was necessary. In most cases, the pattern itself reveals the role relationship of the extracted object. For example, the pattern "murder of <X>" suggests that X is a victim, regardless of whether the pattern actually labels X as a victim or not. Since AutoSlog-TS can not label its extraction patterns automatically, we decided to see whether its unlabeled extraction patterns would be sufficient.

Without semantic features and without extraction pattern labels, the only information we have available about role fillers is the noun phrase itself.[8] To compute statistics on noun phrases, however, would require a gigantic training corpus to produce statistically reliable data. Therefore we use only the head noun of the role filler. Using only the head noun allows us to generalize over different noun phrases, but we may lose some important distinctions captured by the noun modifiers.

Except for the lexical representation of role fillers, the *word-augmented relevancy signatures algorithm* is identical to the feature-augmented relevancy signatures algorithm. We identify relevancy signatures and relevant role fillers by choosing threshold values for R and F (separately for the signatures and role fillers). To classify a new text, we look for the presence of both a relevancy signature and a relevant role filler generated by the same extraction pattern. If we find such a pair, the text is deemed relevant. Otherwise, it is deemed irrelevant.

If we use the thresholds R=.85, F=50 to identify relevant role fillers, we find six relevant nouns: *bomb*, *explosion*, *ELN*, *Medellin*, *assassination*, and *individuals*. These six nouns are highly associated with terrorism[9], but there are only six relevant words so not many texts will be classified as relevant. Without semantic features to generalize over lexical items, the word-augmented approach needs to use lower thresholds to find a significant number of relevant terms. If we lower the thresholds to R=.80, F=20, then we find 41 relevant fillers, which are shown in Figure 6.

Many of the nouns in Figure 6 seem to make sense intuitively. Objects such as car bombs, explosives, assassinations, and the ELN are strongly associated with terrorism. Some of the more general nouns, such as university, hotel, and bus, might not seem to be associated with terrorism. But they are strong indicators of terrorism in combination with a relevancy signature. For example, the bombing of a university, hotel, or bus is very

[8] All of our role fillers are noun phrases because the extraction patterns only extract subjects, direct objects, or objects of prepositional phrases.

[9] Victims of terrorist attacks are often referred to as "individuals" in news reports.

WINDOWS	OQUELI	CAR_BOMB	PIPELINE	NOV_18_89
AUG_18_89	RESIDENCE	BOMB	EXPLOSION	UNIVERSITY
WALKER	ELN	MEDELLIN	JESUITS	ASSASSINATION
NOV_16_89	BODIES	BUILDING	AVENUE	STREET
INDIVIDUALS	PIZARRO	KM	BODYGUARDS	STUDENTS
MUNICIPALITY	UNIFORMS	HOTEL	OCT_31_89	DAMAGE
BOGOTA	BODY	BUS	BENAVIDES	CAR
LOSSES	EXPLOSIVES	MURDER	GALAN	ANTIOQUIA
HEAD				

Figure 6. Relevant role fillers

likely to be an act of terrorism. Figure 6 also contains names of people (e.g., Oqueli, Pizarro, Walker) who were victims or perpetrators of terrorist attacks, location names (e.g., Bogota and Medellin) where terrorism is common, and some incidental nouns (e.g., windows and bodies) that are surprisingly good indicators of terrorism (windows and bodies are rarely mentioned unless something violent happened to them).

While some of these nouns seem more plausible than others, it is important to remember that these nouns must be a role filler for a relevancy signature in order to signal relevance. When both a relevant event expression and a relevant role filler are identified, the combination of words can paint a clear picture of an event.

3.1. FULLY AUTOMATIC TEXT CATEGORIZATION

In Section 1, we discussed text categorization algorithms that achieved good performance in previous experiments with the MUC-4 terrorism corpus. In Sections 2 and 3, we presented two new methods that eliminate the manual knowledge engineering previously required for these algorithms. AutoSlog-TS can generate the extraction patterns automatically, and the word-augmented signatures eliminate the need for semantic features. Putting these pieces together, we can build a NLP-based text categorization system fully automatically. The text categorization system needs only a *preclassified* text corpus as input. AutoSlog-TS can then generate domain-specific extraction patterns and these patterns can be plugged into the NLP system to produce word-augmented relevancy signatures without any manual intervention.

The important question is whether AutoSlog-TS and the word-augmented signatures will produce classification terms that perform well. Our goal is to show that they can produce a text classifier that performs at least as

well as the original algorithms, thereby demonstrating that NLP-based text categorization is possible without relying on hand-crafted domain knowledge. In the next section, we describe a series of experiments to evaluate the AutoSlog-TS extraction patterns and the word-augmented relevancy signatures representation.

4. Experimental results

To evaluate the word-augmented signatures, we focused on two questions: (1) do word-augmented relevancy signatures perform better than relevancy signatures or words alone? (2) do word-augmented relevancy signatures perform at least as well as feature-augmented relevancy signatures? To answer the first question, we compared three text representations: *relevant words*, *relevancy signatures*, and *word-augmented relevancy signatures*. We have already discussed the relevancy signature and augmented relevancy signature representations. The *relevant words* representation refers to individual words without linguistic context. We identify a set of relevant words in the same manner as signatures: we use a training corpus to identify individual words that have a relevance rate and frequency above certain thresholds. These words are called the "relevant words" for a category and a new text is classified as relevant if it contains any of these words. The three representations capture increasing amounts of linguistic context, from individual words to local expressions (signatures) to local expressions plus role filler information (augmented signatures). We evaluated the three text representations using identical classification procedures so that we could cleanly determine the effects of the added linguistic information.

We evaluated performance on four categorization tasks within the MUC-4 corpus. The first category, which we will call *terrorism*, is a broad category covering all texts that mention a relevant terrorist event. The other three categories, *attacks*, *bombings*, and *kidnappings*, are subcategories of terrorism. All of the classification tasks are binary; a text is labeled as relevant or irrelevant with respect to a particular category.[10] Finally, we compared the performance of the word-augmented signatures with the feature-augmented signatures that used hand-coded extraction patterns and semantic features.

4.1. THE TERRORISM CATEGORY

The first category, terrorism, is defined by the MUC-4 guidelines (MUC-4 Proceedings, 1992) as attacks perpetrated against civilians by terrorist

[10] In principle, it would be easy to assign multiple category labels to a text by running all of the classifiers sequentially.

groups or organizations.[11] This category is the most coarse because all of the subcategories fall under this broad definition of terrorism. Our classification experiments involve both a training and a testing phase for each category. The 1500 MUC-4 development texts were used as the training corpus and the 200 MUC-4 test documents were used as a blind test set for all the categories. 51% of the MUC-4 texts are relevant to the general terrorism category. The experiments were done as follows:

1. We ran AutoSlog-TS over the training corpus, which produced a set of extraction patterns for the terrorism domain. The same set of extraction patterns was used for all the categories.
2. We loaded the extraction patterns with frequency > 1 into CIRCUS and reprocessed the training corpus. For each text, we kept track of the extraction patterns that were activated and the role fillers that were extracted by each pattern.
3. For each extraction pattern and role filler, we computed its relevance rate and overall frequency in the training corpus. The relevance rate depends on the category being tested (it is a conditional probability that a text is relevant to a given category). We also computed the relevance rate and frequency of the individual words in the corpus for the relevant words representation.
4. We ran the three classification algorithms using a variety of threshold values. For the relevant words and relevancy signatures, we tested R = 10 to 100 by increments of 5 and F = 3 to 50 by increments of 2. For the word-augmented signatures, we need two sets of threshold values, one for the signatures and one for the fillers. For the signatures, we tested R = 10 to 100 by increments of 5, F = 3, and F = 5 to 50 by increments of 5. For the fillers, we tested R = 10 to 100 by increments of 5, F = 3, and F = 5 to 25 by increments of 5. We increased the increments for the word-augmented signatures because of the combinatorics of varying four thresholds.

Because we tested so many threshold values, each algorithm produced many data points. We plotted only the data points that showed the best performance for each representation. In a real application, one must choose a set of threshold values in advance and it is often difficult to know what values are best. In previous work, we developed a procedure for identifying good threshold values from the training corpus empirically (Riloff and Lehnert, 1994), but we do not use that procedure here because we want to see what each representation can do in the best case.

[11] The MUC-4 definition specifies some additional constraints, for example only terrorist attacks in Latin America are relevant.

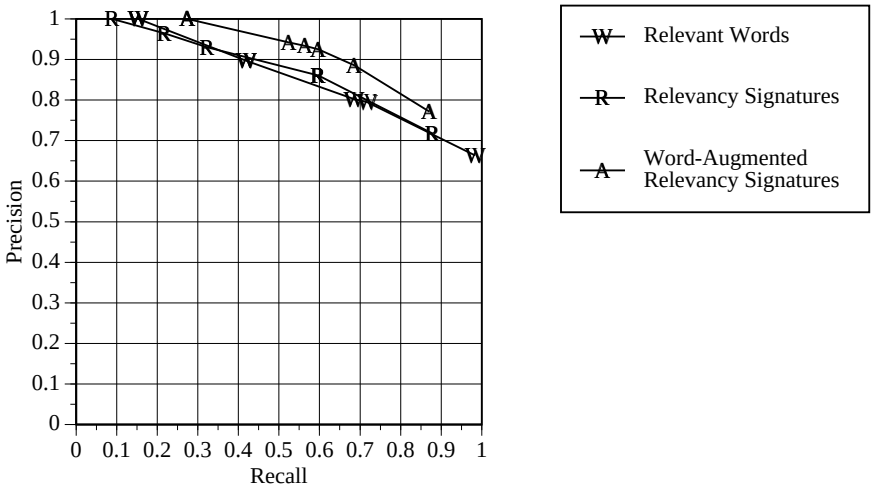
For each algorithm, we plotted the 7 best data points using the F-measure as our evaluation metric. The F-measure is computed as: $F(\beta) = \frac{(\beta^2 + 1.0) \times P \times R}{\beta^2 \times P + R}$, where P is precision and R is recall. Recall is the ratio of texts correctly classified as relevant over the total number of relevant texts. Precision is the ratio of texts correctly classified as relevant over the total number of texts classified as relevant. The F-measure combines recall and precision into a single measure using a β value to adjust their relative weighting. Our approach is geared toward high-precision classification so we used the following five β values: 1.0, 0.5, 0.3, 0.2, 0.1. β=1.0 gives equal weighting to recall and precision, β=0.5 gives precision twice as much weight as recall, and the lower β values give increasing weight to precision. We also computed an absolute *Precision* measure, which is the highest precision value achieved at any recall level.

Figure 7 shows the results of the three text classification algorithms on the general terrorism category. 124 of the 200 test documents are relevant to this category, so a trivial algorithm that classifies every text as relevant would achieve 62% precision. The most striking result is that the word-augmented relevancy signatures achieved 27% recall with 100% precision. Relevancy signatures performed nearly as well, achieving 22% recall with 96% precision. Relevant words had a tougher time at the high precision end, achieving only 15% recall with 100% precision. Since the focus of our work is on high-precision categorization, this end of the spectrum is most important to us. While some applications will demand high recall, many applications are more concerned with high precision.[12] Our results demonstrate that extraction-based categorization can perform high-precision classification at higher recall levels than the word-based algorithm.

Figure 7 shows that the word-augmented signatures consistently performed better than the other representations. The only disadvantage of the signature-based algorithms is that recall appears to be capped at about 90%. The richer signature representations make it less likely that relevant texts will contain a relevancy signature. On the other hand, the relevant words algorithm achieves 98% recall but with only 66% precision, which means that it is classifying virtually every text as relevant. In contrast, the word-augmented signatures were able to obtain nearly 70% recall with 90% precision.

It is interesting to observe that the word-augmented signatures achieved almost twice the recall that the relevant words representation achieved at 100% precision (27% vs. 15% recall). At first glance, this seems counterin-

[12] With large text databases, even 1% of the relevant texts may be more than a user can digest. In this type of situation, the most important thing is to reduce the number of irrelevant texts that the user must wade through. Of course, the more relevant texts than can be accurately identified, the better.



Algorithm	F(1)		F(.5)		F(.3)		F(.2)		F(.1)		Prec.	
Rel Words	.98	.66	.72	.79	.42	.90	.42	.90	.15	1.0	.15	1.0
Rel Sigs	.88	.72	.60	.86	.60	.86	.32	.93	.22	.96	.09	1.0
Word-aug. Sigs	.87	.77	.69	.89	.56	.93	.52	.94	.27	1.0	.27	1.0

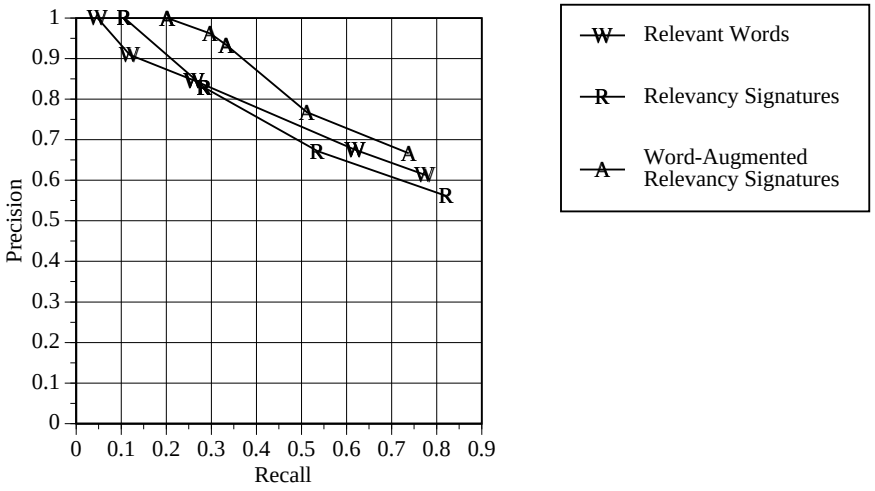
Figure 7. Terrorism category results

tuitive because the word-augmented signatures are a richer representation which should adversely affect recall. The explanation lies in the threshold values. Since relevant words are such a simple representation, the relevance rate threshold must be cranked up very high to identify reliable classification terms. The data point that achieved 15% recall with 100% precision used the threshold R=.95. Word-augmented signatures are a richer representation, so reliable classification terms can be identified with lower threshold values. The data point that achieved 27% recall with 100% precision used the threshold R=.75 for the signatures and R=.80 for the fillers. With lower threshold values, more relevancy signatures and relevant role fillers can be found which allows more texts to be classified as relevant.

4.2. THE ATTACK CATEGORY

The attack subcategory involves texts that describe a generic terrorist attack or murder involving weapons other than bombs. 499 of the 1500 training texts (33%) mention terrorist attacks, and 84 of the 200 test texts (42%)

mention attacks. Figure 8 shows the results of the three text representations on the attack category.



Algorithm	F(1)		F(.5)		F(.3)		F(.2)		F(.1)		Prec.	
Rel Words	.77	.61	.62	.68	.26	.85	.26	.85	.12	.91	.05	1.0
Rel Sigs	.82	.56	.54	.67	.29	.83	.29	.83	.11	1.0	.11	1.0
Word-aug. Sigs	.74	.67	.51	.77	.33	.93	.30	.96	.20	1.0	.20	1.0

Figure 8. Attack category results

As before, the word-augmented signatures consistently performed better than the other two representations. At the high precision end, word-augmented signatures achieved 20% recall with 100% precision, which was nearly double the recall achieved by relevancy signatures (11%) and four times the recall obtained by relevant words (5%). The word-augmented signatures were also able to achieve 30% recall with 96% precision, which is nearly 10 points higher in precision than either of the other representations could achieve at similar recall levels.

Some interesting insights can be gained by looking at the relevant words, signatures, and role fillers that were identified for the attack category. The relevant words and relevancy signatures used to produce the F(.1) column in Figure 8 are:

Relevant Words: (R=.90, F=11)

porth	gunmen	tojeira	wolff	galdamez
shot_to_death	ellacuria	rector	navas	jesuits
pizarro's	disabled	priests'	tonacatepeque	valencia

Relevancy Signatures: (R=.75, F=25)

<subject> was shot_to_death death of <pp>

was murdered on <pp> <subject> was assassinated

The most striking observation is that there are 15 relevant words but only 4 relevancy signatures. Both representations produced roughly the same recall (11-12%), but the relevancy signatures were more accurate, obtaining 100% precision compared with 91% for the relevant words. We can see that the relevancy signatures are more general than the words. Expressions such as "<X> was murdered" and "death of <X>" are likely to occur in many attack descriptions. The relevant words are more specific. Many of them are proper names of people who were attacked (e.g., *Wolff*, *Ellacuria*, and *Valencia*) or references to clergymen who were murdered (*jesuits, priests'*, *rector*). We conclude that the relevant words algorithm was successful at identifying terms that were associated with specific attack incidents, but that the relevancy signatures algorithm was more successful at identifying general expressions associated with attacks.

When we look at the word-augmented signatures, we find that the role fillers did most of the work. In the F(.1) column, the algorithm used thresholds of R=.60 and F=3 for the signatures, which yielded 889 attack signatures. Most of these expressions are not specific to attacks. For the role fillers, thresholds of R=.70 and F=5 were used, which produced 131 attack fillers. Together, these proved to be a powerful combination resulting in 100% precision with 20% recall. Interestingly, to improve recall the algorithm lowered the relevance threshold for signatures but increased the frequency threshold for role fillers. Consequently, the role fillers took on increased responsibility for identifying relevant expressions. For the F(.2) column, thresholds of R=.40 and F=3 produced 2092 attack signatures and thresholds of R=.70 and F=15 produced the 23 role fillers below:

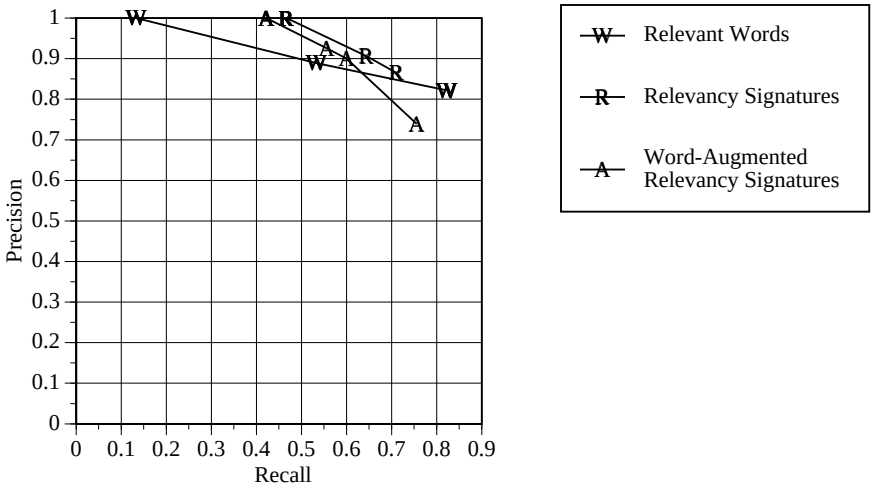
ellacuria	walker	jesuits	nov_16_89	ungo
pizarro	bullets	residence	aug_18_89	university
bodyguards	assassination	uniforms	priests	murder
witness	massacre	murderers	victim	corps
galan	crime	mar_19_89		

Some of these words are very specific, referring to particular people and dates associated with attacks. But many of the words are quite general, such as *bullets*, *assassination*, *murder*, and *victim*. Remember that the relevant

words algorithm did not perform well on this category, so individual words by themselves are not enough to signal relevance. But these words can be compelling as role fillers. For example, "masterminding an assassination", "claiming responsibility for an attack", or "identifying a victim" are all expressions that strongly suggest an attack.

4.3. THE BOMBING CATEGORY

The bombing category involves texts that describe a bombing incident perpetrated by terrorists. 223 of the 1500 training texts (15%) mention relevant bombing incidents, and 45 of the 200 test texts (22.5%) mention bombings. Figure 9 shows the results of the three text representations on the bombing category.



Algorithm	F(1)		F(.5)		F(.3)		F(.2)		F(.1)		Prec.	
Rel Words	.82	.82	.82	.82	.53	.89	.53	.89	.13	1.0	.13	1.0
Rel Sigs	.71	.86	.64	.91	.47	1.0	.47	1.0	.47	1.0	.47	1.0
Word-aug. Sigs	.76	.74	.60	.90	.42	1.0	.42	1.0	.42	1.0	.42	1.0

Figure 9. Bombing category results

This category behaved differently than the terrorism and attack categories. The relevancy signatures performed best, doing slightly better than the word-augmented signatures. The relevant words did substantially worse than both of the signature-based representations. Focusing again on the

F(.1) column, we can understand the behavior when we look at the relevancy signatures and relevant words that were used.

Relevant Words: (R=1.0, F=7)

blast blown_up circulate mar_26_90 seamen
sendero_luminoso vina

Relevancy Signatures: (R=.90, F=5)

<subject> damaged windows	<subject> was detonated
<subject> circulate	<subject> shattered windows
<subject> was blown_up	<subject> went_off
<subject> was dynamited	<subject> was set_off
<subject> shattered	shattered <dobj>
was injured <dobj>	hurled <dobj>
escaped <dobj>	went_off at <pp>
dynamite_attack on <pp>	exploded in_front_of <pp>
went_off on <pp>	exploded near <pp>
hurled at <pp>	pipeline in <pp>
explosion in <pp>	bomb at <pp>
exploded at <pp>	

Many more relevancy signatures were found than relevant words for the bombing category. This is partly because the relevant words algorithm had to crank the relevance threshold up to R=1.0 in order to get 100% precision. The relevancy signatures algorithm, using a slightly richer representation, could achieve 100% precision with R=.90. But this is not the whole story: all of the relevancy signatures except three also had a relevance rate of 1.0. There are many more signatures than words with a relevance rate of 1.0 because the signatures can distinguish between different contexts involving the same word.

For example, Figures 10 and 11 show all of the signatures triggered by the words *detonated* and *exploded*. The *detonated* signatures are revealing because we see that some of them have a much higher relevance rate than others. For example, the passive form of detonated (<subject> was detonated) had a relevance rate of 1.0, while the active form (<subject> detonated) had a relevance rate of only .75. Only one of the six *detonated* signatures (<subject> was detonated) had a high enough relevance rate and frequency to become a relevancy signature for the bombing category.

The *exploded* signatures tell a similar story. Only the top two *exploded* signatures passed the thresholds of R=.90 and F=5 to become relevancy signatures. The high frequencies associated with some of the signatures sug-

Signature	Rel	Freq
<subject> was detonated	1.0	12
was detonated on <pp>	1.0	3
was detonated in <pp>	1.0	3
detonated <dobj>	.78	9
<subject> detonated	.75	12
detonated by <pp>	.50	2

Figure 10. Bombing signatures with trigger word *detonated*

gest that many occurrences of the word *exploded* did not describe terrorist attacks. But when an object explodes *near*, *in_front_of*, or *inside* something, then the likelihood of a bombing incident is considerably higher. The bombing category nicely illustrates how the signature representation can be more powerful than individual words.

Signature	Rel	Freq
exploded near <pp>	1.0	7
exploded in_front_of <pp>	1.0	7
exploded inside <pp>	1.0	4
exploded during <pp>	1.0	2
was exploded on <pp>	1.0	1
exploded at <pp>	.90	21
<subject> exploded	.87	113
exploded <dobj>	.87	15
exploded on <pp>	.86	36
exploded in <pp>	.85	54
<subject> was exploded	.75	4
was exploded in <pp>	.50	2

Figure 11. Bombing signatures with trigger word *exploded*

The word-augmented signatures did not perform quite as well as relevancy signatures, but both signature representations performed considerably better than the relevant words. In the bombing domain, the role fillers did not improve performance over the signatures. When we look at the threshold values that produced the F(.1) data point, we see that both signature representations used the values R=.90 and F=5 to select relevancy signatures. So both algorithms used exactly the same bombing signatures. The word-augmented signatures set the role filler thresholds at R=.10 and F=3, which were the minimum possible threshold values. Presumably, the

algorithm would have dropped the thresholds all the way down to zero if we had permitted it, which would have made the word-augmented signatures behave exactly the same as relevancy signatures.

The bombing category demonstrates that sometimes the relevancy signatures by themselves are capable of achieving good performance and the role filler information is not necessary. Bombing incidents are often the result of terrorism, so any reference to a bombing has a good chance of being relevant.

4.4. THE KIDNAPPING CATEGORY

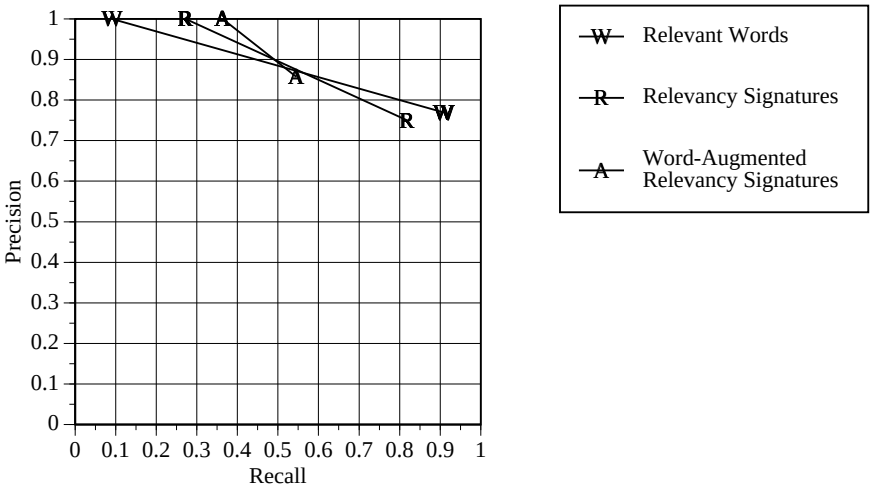
The kidnapping category involves texts that describe a kidnapping event perpetrated by terrorists. 92 of the 1500 training texts (6%) mention relevant kidnapping incidents, and 11 of the 200 test texts (5.5%) mention kidnappings. The kidnapping category contained considerably fewer relevant texts than the other categories, making it potentially more challenging because there were fewer than 100 relevant texts available for training. Figure 12 shows the results of the three text representations.

The word-augmented signatures clearly achieved the best precision. Due to the small number of relevant texts in the test set, each algorithm produced only two distinct data points. All three representations were able to achieve 100% precision, but with varying levels of recall: word-augmented signatures achieved 36% recall, relevancy signatures achieved 27% recall, and the relevant words achieved only 9% recall. The word-augmented signatures were able to maintain good precision with higher recall, obtaining 55% recall with 86% precision, but they were not able to achieve greater recall. The other two representations got much higher recall, but at the expense of precision.

The kidnapping category is especially interesting for several reasons. First, remember that only 5.5% of the test documents were relevant so all of the algorithms were able to sift through the test collection and find several relevant documents. Second, the recall and precision results are comparable to the results for the other categories, which is impressive considering how few kidnapping texts (92) were available for training. This provides evidence that the signature representations are capable of achieving good performance even when only a small number of relevant documents are available for training.

As before, it is interesting to look at the words and signatures used by the algorithms. The relevant words and relevancy signatures used for the F(.1) column are:

Relevant Words: (R=.85, F=5)



Algorithm	F(1)		F(.5)		F(.3)		F(.2)		F(.1)		Prec.	
Rel Words	.91	.77	.91	.77	.91	.77	.91	.77	.09	1.0	.09	1.0
Rel Sigs	.82	.75	.82	.75	.27	1.0	.27	1.0	.27	1.0	.27	1.0
Word-aug. Sigs	.55	.86	.55	.86	.36	1.0	.36	1.0	.36	1.0	.36	1.0

Figure 12. Kidnapping category results

leslie amico azurdia echevarria berrocal
olson castillo's castellar diniz heyndal
aeu abducted parker impresario

Relevancy Signatures: (R=.85, F=15)
was kidnapped in <pp> was kidnapped on <pp>

The relevant words are mostly proper names associated with specific kidnapping incidents. While there are only two kidnapping signatures, they represent general expressions associated with kidnappings. It is interesting that only two signatures were able to achieve 36% recall, although this translates to just four texts because there were only 11 relevant documents. But there is a general principle at work here: a few general expressions will often have wide applicability. If the text categorization algorithms can find a small set of reliable but general signatures, then those few signatures may be found in a significant number of relevant documents.

The word-augmented signatures performed well on the kidnapping category but behaved differently than in the attack category. For attacks, the

best performance resulted from a large set of general signatures combined with a relatively small set of relevant role fillers. For kidnappings, the best performance resulted from a small set of kidnapping signatures combined with a relatively large set of general role fillers. For the kidnapping category, signature thresholds of R=.30[13] and F=25 produced the eight kidnapping signatures below:

was kidnapped in <pp>	<subject> kidnapped
<subject> release	was kidnapped on <pp>
<subject> was kidnapped	kidnapping <dobj>
was kidnapped by <pp>	kidnapped <dobj>

All of these signatures are strongly associated with kidnapping. Thresholds of R=.15 and F=10 identified 95 role fillers for the kidnapping category, which consisted of specific people and places as well as general words associated with kidnappings, such as "kidnappers" and "hostages."

4.5. COMPARING AUTOMATIC AND HAND-CRAFTED DICTIONARIES

Finally, we conducted experiments to compare the performance of the new word-augmented signatures with the original feature-augmented signatures. There are two important variables: (1) the word-augmented signatures used extraction patterns generated by AutoSlog-TS, while the feature-augmented signatures used hand-crafted extraction patterns, and (2) the word-augmented signatures used lexical items to represent role fillers while the feature-augmented signatures used semantic features.

To factor out the effect of the AutoSlog-TS dictionary, we evaluated the relevancy signatures algorithm using both the AutoSlog-TS extraction patterns and the hand-crafted extraction patterns. If the relevancy signatures algorithm performs similarly using both dictionaries, then we can conclude that any subsequent improvement by the word-augmented signatures must be due to the role fillers.

Figure 13 shows the results of relevancy signatures and word-augmented signatures for the terrorism category using both the AutoSlog-TS and hand-crafted dictionaries. First, relevancy signatures perform nearly the same using both dictionaries. This result suggests that the AutoSlog-TS dictionary duplicates most of the functionality of the hand-crafted dictionary. Second, both types of augmented signatures outperform the relevancy signatures alone. This implies that representing role fillers improves performance in

[13] Remember that kidnappings make up only 6% of the training corpus, so a 30% correlation with kidnapping texts is substantially greater than one would expect at random. It is important to consider the percentage of relevant texts in the training set when selecting threshold values.

the terrorism domain. Third, the feature-augmented signatures perform slightly better than the word-augmented signatures, but the difference is small. This result indicates that lexical role fillers can approximate the functionality of semantic features for the purposes of text categorization.

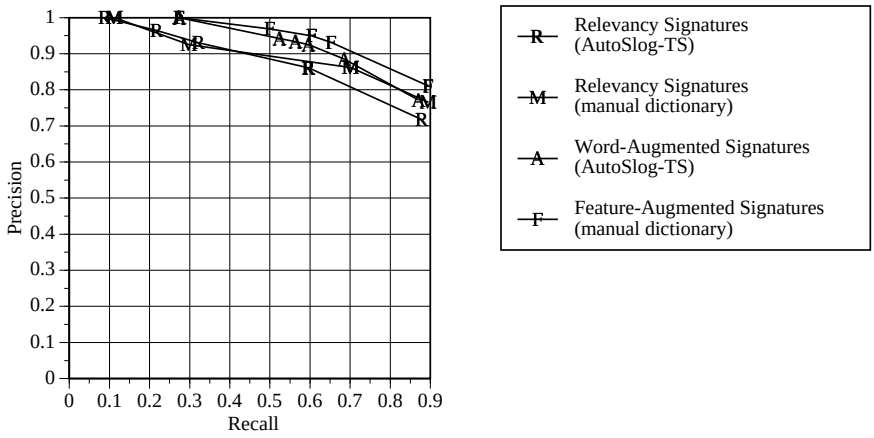


Figure 13. Comparing dictionaries and representations for terrorism

We ran the same experiments on the attack, bombing, and kidnapping categories. The results were similar across the categories. The only exception was the bombing category, in which the relevancy signatures based on the AutoSlog-TS dictionary performed substantially better then the relevancy signatures based on the hand-crafted dictionary. We conclude that AutoSlog-TS generated some extraction patterns that were strongly associated with bombings but were not defined in the hand-crafted dictionary. This illustrates the potential benefits of corpus-based knowledge acquisition. By gathering data from a training corpus automatically, AutoSlog-TS can identify expressions that are common in the domain but that a human system developer might not consider.

5. Conclusions

Text categorization was one of the earliest areas of information retrieval research (e.g., (Maron, 1961; Borko and Bernick, 1963)). Consequently, most categorization systems use statistical IR models (e.g., (Hoyle, 1973; Stanfill and Waltz, 1986; Fuhr *et al.*, 1991)). But some text categorizations systems have been built that exhibit strong performance using hand-coded knowledge bases (e.g., (Hayes and Weinstein, 1991; Goodman, 1991)). Our approach represents a middle ground that aims to benefit from domain

knowledge but also strives to acquire the necessary domain knowledge automatically.

We believe that our text categorization algorithms represent a new approach to integrating natural language processing techniques with information retrieval applications. For one, we use information extraction techniques to support syntactic processing, concept extraction, and robust sentence analysis. Second, the AutoSlog-TS system and the text categorization algorithms allow us to generate complex classification terms automatically. The word-augmented signatures capture linguistic context that is much richer than individual words or phrases. Putting these pieces together, we can build a text classifier that recognizes domain-specific expressions and role relationships using natural language processing capabilities. Furthermore, the system can be easily retrained for new categories with no manual knowledge engineering. The system is fully portable across domains given only a training corpus of preclassified texts. We believe that this represents an important step toward practical applications of conceptual natural language processing to information retrieval tasks.

6. Acknowledgments

This research was funded by NSF grant IRI-9509820 and NSF grant MIP-9023174.

References

Borko, H. and Bernick, M. 1963. Automatic Document Classification. *J. ACM* 10(2):151–162.

Fuhr, N.; Hartmann, S.; Lustig, G.; Schwantner, M.; and Tzeras, Konstadinos 1991. AIR/X - A Rule-Based Multistage Indexing System for Large Subject Fields. In *Proceedings of RIAO 91*. 606–623.

Goodman, M. 1991. Prism: A Case-Based Telex Classifier. In *Proceedings of the Second Annual Conference on Innovative Applications of Artificial Intelligence*. AAAI Press. 25–37.

Hayes, Philip J. and Weinstein, Steven P. 1991. Construe-TIS: A System for Content-Based Indexing of a Database of News Stories. In *Proceedings of the Second Annual Conference on Innovative Applications of Artificial Intelligence*. AAAI Press. 49–64.

Hoyle, W. 1973. Automatic Indexing and Generation of Classification Systems by Algorithm. *Information Storage and Retrieval* 9(4):233–242.

Huffman, S. 1996. Learning information extraction patterns from examples. In Wermter, Stefan; Riloff, Ellen; and Scheler, Gabriele, editors 1996, *Connectionist, Statistical, and Symbolic Approaches to Learning for Natural Language Processing*. Springer-Verlag, Berlin. 246–260.

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. 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.

Maron, M. 1961. Automatic Indexing: An Experimental Inquiry. *J. ACM* 8:404–417.

MUC-3 Proceedings, 1991. *Proceedings of the Third Message Understanding Conference (MUC-3).* Morgan Kaufmann, San Mateo, CA.

MUC-4 Proceedings, 1992. *Proceedings of the Fourth Message Understanding Conference (MUC-4).* Morgan Kaufmann, San Mateo, CA.

MUC-5 Proceedings, 1993. *Proceedings of the Fifth Message Understanding Conference (MUC-5).* Morgan Kaufmann, San Francisco, CA.

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. 1994. *Information Extraction as a Basis for Portable Text Classification Systems.* Ph.D. Dissertation, CMPSCI Technical Report 95-04, Department of Computer Science, University of Massachusetts, Amherst, MA.

Riloff, E. 1995. Little Words Can Make a Big Difference for Text Classification. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.* 130–136.

Riloff, E. 1996a. An Empirical Study of Automated Dictionary Construction for Information Extraction in Three Domains. *Artificial Intelligence* 85:101–134.

Riloff, E. 1996b. Automatically Generating Extraction Patterns from Untagged Text. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence.* The AAAI Press/MIT Press. 1044–1049.

Salton, G., editor 1971. *The SMART Retrieval System: Experiments in Automatic Document Processing.* Prentice Hall, Englewood Cliffs, NJ.

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.

Stanfill, C. and Waltz, D. 1986. Toward Memory-Based Reasoning. *Communications of the ACM* 29(12):1213–1228.

Turtle, Howard and Croft, W. Bruce 1991. Efficient Probabilistic Inference for Text Retrieval. In *Proceedings of RIAO 91.* 644–661.