To appear in *AI Journal*, Vol. 85, Elsevier Publishers, cover date August 1996.

An Empirical Study of Automated Dictionary Construction for Information Extraction in Three Domains

Ellen Riloff

Department of Computer Science, University of Utah, Salt Lake City, UT 84112

A primary goal of natural language processing researchers is to develop a knowledge-based natural language processing (NLP) system that is portable across domains. However, most knowledge-based NLP systems rely on a domain-specific dictionary of concepts, which represents a substantial knowledge-engineering bottleneck. We have developed a system called AutoSlog that addresses the knowledge-engineering bottleneck for a task called *information extraction*. AutoSlog automatically creates domain-specific dictionaries for information extraction, given an appropriate training corpus. We have used AutoSlog to create a dictionary of extraction patterns for terrorism, which achieved 98% of the performance of a hand-crafted dictionary that required approximately 1500 person-hours to build. In this paper, we describe experiments with AutoSlog in two additional domains: joint ventures and microelectronics. We compare the performance of AutoSlog across the three domains, discuss the lessons learned about the generality of this approach, and present results from two experiments which demonstrate that novice users can generate effective dictionaries using AutoSlog.

1 Introduction

Portability is a crucial concern for researchers in knowledge-based natural language processing (NLP). Knowledge-based NLP systems typically rely on a conceptual dictionary that has been manually encoded for a specific domain. Although knowledge-based systems have performed well on certain tasks (e.g., [2,4,5,11,16,23]), these systems will not be practical for real world applications until the knowledge that they need can be acquired automatically.

We have developed a system called AutoSlog that generates conceptual dictionaries for information extraction automatically. Information extraction (IE) is essentially a form of text skimming, in which specific types of information are extracted from text. There has been a lot of work recently on information extraction in conjunction with the recent message understanding conferences [26–28]. Most information extraction systems rely on a manually encoded dictionary of extraction patterns (e.g., see [12,15,1]). Using AutoSlog, the UMass/MUC-4 system was the first system that could acquire domain-specific extraction patterns automatically [17,18].

In previous work, we showed that AutoSlog could create effective extraction patterns for the domain of terrorism [30]. A dictionary generated by AutoSlog for the terrorism domain achieved 98% of the performance of a hand-crafted dictionary that required approximately 1500 person-hours to build. The heuristics used by AutoSlog are domain-independent linguistic rules, but it was unclear whether these heuristics would be effective in other domains. In this paper, we describe the results of experiments with AutoSlog in two additional domains: joint ventures and microelectronics. Our goal was to determine whether the domain-independent linguistic rules used by AutoSlog are sufficient to generate effective extraction patterns for other types of domains. If not, would small modifications to the heuristics be sufficient to produce good dictionaries? Or did the heuristics need to be completely overhauled? Or perhaps this domain-independent approach was not portable at all.

We also conducted two experiments to determine whether novice users could produce effective dictionaries using AutoSlog. Knowledge acquisition systems that can be used only by computer scientists will not be practical in most real-world situations. The results of these experiments provided valuable feedback about the effectiveness and variation of dictionaries produced by different people.

In the first section, we provide some background about information extraction and give a brief overview of the CIRCUS sentence analyzer used in these experiments. In Section 2, we describe the AutoSlog system for automated dictionary construction, and present results from the terrorism domain. In Section 3, we describe the modifications made to AutoSlog and experimental results for the joint ventures and microelectronics domains. Section 4 describes the experiments with novice users. Finally, Section 5 discusses related work and the implications of AutoSlog.

1.1 Information Extraction

Information extraction (IE) is a natural language processing task that involves

automatically extracting specific types of information from text. In contrast to in-depth understanding, information extraction systems extract only the information that is relevant to a specific domain. For example, an information extraction system for the domain of terrorism might extract the names of perpetrators, victim, physical targets, and weapons involved in a terrorist incident. An information extraction system for the domain of joint ventures might extract the names of people and companies involved in joint ventures and the names of products and facilities associated with them.

Information extraction has received a lot of attention recently because of the message understanding conferences (MUCs) sponsored by the U.S. government [26–28]. The message understanding conferences are competitive performance evaluations that involve participants from a variety of academic and industrial research labs. The third and fourth message understanding conferences (MUC-3 and MUC-4) were held in 1991 and 1992 and involved information extraction for the domain of Latin American terrorism. Each participating site developed an information extraction system for the terrorism domain, and the systems were formally evaluated and compared. Fifteen sites participated in MUC-3 and seventeen sites participated in MUC-4. The fifth message understanding conference (MUC-5) was held in 1994 and involved information extraction for two new domains: joint ventures (a business domain) and microelectronics (a technical domain).

The information extraction task was to extract relevant information from texts and put the extracted information into predefined templates. For MUC-4, 22 types of information had to be extracted for each terrorist incident mentioned in a text. Figure 1 shows a text from the MUC-4 corpus that describes a bombing of the U.S. embassy in Miraflores, Peru. For this text, a bombing template had to be generated that included the date of the bombing ("15 JANUARY"), the location ("MIRAFLORES"), the perpetrators ("TEN TERRORISTS"), the weapons ("DYNAMITE STICKS"), the physical target ("U.S. EMBASSY FACILITIES"), the human targets ("EMBASSY OFFICIALS" and "SECURITY OFFICERS"), and the information about damage and casualties.

The MUC participants were provided with a development corpus to use for training purposes and a blind test set for the final evaluation. The MUC-4 development corpus consisted of 1500 texts and associated answer keys. The answer keys are templates that were filled out manually with the information that should be extracted from the texts. If several terrorist incidents were reported in a text, then multiple templates had to be filled out. If no terrorist incidents were reported, then no templates had to be filled out. 53% of the texts in the MUC-4 corpus contained relevant information and therefore had one or more associated answer key templates.

LIMA, 16 JAN 90 (TELEVISION PERUANA) – [TEXT] TEN TERRORISTS HURLED DYNAMITE STICKS AT U.S. EMBASSY FACILITIES IN THE MIRAFLORES DISTRICT, CAUSING SERIOUS DAMAGE BUT FORTUNATELY NO CASUALTIES. THE ATTACK TOOK PLACE AT 2100 ON 15 JANUARY [0100 GMT ON 16 JAN].

INSIDE THE FACILITY, WHICH WAS GUARDED BY 3 SECURITY OFFICERS, A GROUP OF EMBASSY OFFICIALS WERE HOLDING A WORK MEETING.

ACCORDING TO THE FIRST POLICE REPORTS, THE ATTACK WAS STAGED BY 10 TERRORISTS WHO USED 2 TOYOTA CARS WHICH WERE LATER ABANDONED. ONE OF THE VEHICLES WAS LEFT ON THE THIRD BLOCK OF JOSE PARDO AVENUE, WHILE THE OTHER WAS LEFT ON THE FIRST BLOCK OF BELLA VISTA STREET IN MIRAFLORES.

Fig. 1. A MUC-4 terrorism text

1.2 The CIRCUS Sentence Analyzer

The natural language processing group at the University of Massachusetts participated in MUC-3, MUC-4, and MUC-5 using a conceptual sentence analyzer called CIRCUS [16]. The heart of CIRCUS is a domain-specific dictionary of *concept nodes*. A concept node is essentially a case frame that is activated by certain linguistic expressions and extracts information from the surrounding text. Figure 2 shows a sample sentence and an instantiated concept node produced by CIRCUS. The concept node $MURDER-PASSIVE$ is activated by the passive form of the verb "murdered" and extracts the "three peasants" as victims and the "guerrillas" as perpetrators.

Sentence: Three peasants were murdered by guerrillas.

$MURDER-PASSIVE$
victim = "three peasants"
perpetrator = "guerrillas"

Fig. 2. An instantiated concept node

Figure 3 shows the concept node definition of $MURDER-PASSIVE$ in the dictionary. This concept node is activated by passive forms of the verb "murdered", such as "was murdered", "were murdered", and "have been murdered." Once activated, it extracts the subject of the verb as a victim, and the object of the preposition "by" as a perpetrator. The dictionary also contains a similar concept node called $MURDER-ACTIVE$ which is activated by active forms of the verb "murdered", such as "John murdered Sam" or "John has murdered Sam." $MURDER-ACTIVE$ extracts the subject of the verb as a perpetrator (i.e., John) and its direct object as a victim (i.e., Sam).

A concept node definition contains a trigger word that determines when the concept node is activated. For example, both $MURDER-PASSIVE$ and $MURDER-

Name:	$MURDER-PASSIVE$
Trigger Word:	murdered
Variable Slots:	(victim (*SUBJECT* 1))
	(perpetrator (*PREP-PHRASE* (is-prep? '(by))))
Slot Constraints:	(class VICTIM *SUBJECT*)
	(class PERPETRATOR *PREP-PHRASE*)
Constant Slots:	(type murder)
Enabling Conditions:	(passive)

Fig. 3. The concept node definition for $MURDER-PASSIVE$

ACTIVE$ are triggered by the word "murdered." However, a concept node stays active only if its enabling conditions are satisfied. The enabling conditions ensure that each concept node recognizes specific linguistic expressions. For example, $MURDER-PASSIVE$ contains enabling conditions that recognize passive forms of the verb "murdered", and $MURDER-ACTIVE$ contains enabling conditions that recognize active forms of the verb "murdered." Only one of these concept nodes will remain active for each occurrence of the verb "murdered."

A concept node definition also contains variable slots that identify the syntactic constituents extracted by the concept node and their role assignments (e.g., victim or perpetrator). Slot constraints restrict the kind of fillers that a slot will accept (e.g., the victim slot only accepts humans). Each concept node also has a constant slot that defines the event type represented by the concept node. For example, both of the murder concept nodes have the type "murder" because they are activated by expressions that refer to murder.

All of the information extraction done by CIRCUS happens through concept nodes, so it is essential to have a concept node dictionary that provides good coverage of the domain. The UMass/MUC-3 system [19] used a concept node dictionary for the terrorism domain that was constructed by hand. Although the hand-crafted dictionary performed well[1], we estimate that it required approximately 1500 person-hours to build. Furthermore, creating concept nodes by hand required system developers who were experienced with CIRCUS. As a result, the UMass/MUC-3 system was not portable across domains. To apply the system to a new domain, the entire knowledge engineering process had to be repeated.

[1] The UMass/MUC-3 system had the highest combined recall and precision of all the MUC-3 systems [20].

2 Automated Dictionary Construction Using AutoSlog

2.1 Motivation

Building a concept node dictionary by hand was tedious and time-consuming, but in retrospect we realized that the process mainly involved looking for gaps in the dictionary and then creating definitions to fill those gaps. Looking back, most concept nodes were defined using this four-step procedure:

(1) Run a text through CIRCUS and identify information that should have been extracted but was not (the "targeted" information).
(2) Determine whether the targeted information was the subject of a clause, the direct object, or a prepositional phrase.
(3) Determine which word in the sentence was the strongest indicator that the information should have been extracted. Use this word as the trigger word for a concept node.
(4) Create a concept node that is activated by the trigger word in the same immediate context, and extracts information from the syntactic constituent identified in step (2).

On the surface, Step (3) seems like the most difficult step to automate. However, in most cases the trigger word can be reliably identified using simple linguistic rules. For example, if the targeted information is the subject or direct object of a verb, then the verb is usually an appropriate trigger word. If the targeted information is in a prepositional phrase, then a pp-attachment algorithm can be used to find the best trigger word. Simple rules also determine how much context should be included in Step (4). In general, the concept node should be activated by the same word in the same type of immediate linguistic context (e.g., active or passive verb forms).

Based on these observations, we developed a system that uses linguistic rules to build concept node definitions automatically. The advantages of automating this process are (1) a substantial reduction in the time required for knowledge engineering and (2) a dictionary that potentially provides better coverage of the domain. The next section describes the AutoSlog system that automatically creates concept node dictionaries using this approach. The following section presents the results of an experiment with AutoSlog in the MUC-4 terrorism domain.

The main idea behind AutoSlog is that domain-independent linguistic rules can be used to construct patterns for information extraction automatically. As input, AutoSlog needs examples of information that should be extracted. Figure 4 shows a flowchart that depicts the stages involved in automated dictionary construction.

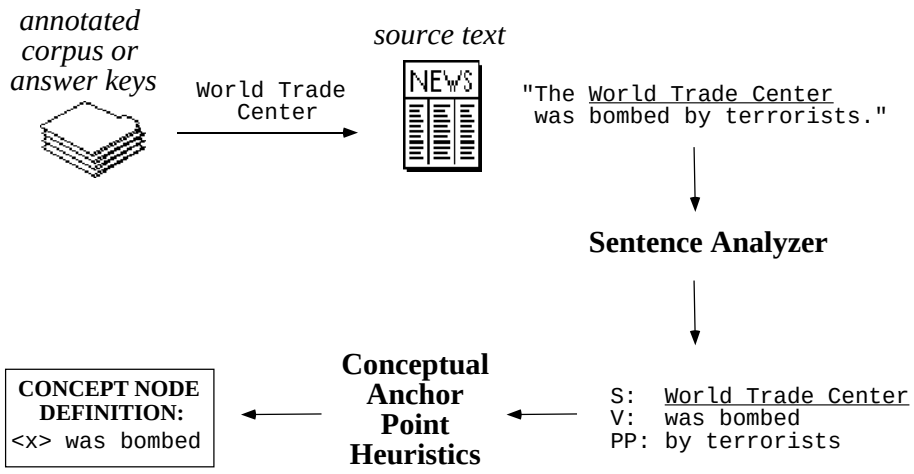

Fig. 4. AutoSlog flowchart

STEP 1: GENERATE AN APPROPRIATE TRAINING CORPUS

The input to AutoSlog is a set of answer keys or an annotated corpus in which the targeted information for each text has been labeled with semantic tags. The only requirement imposed by AutoSlog is that only noun phrases can be tagged. To illustrate, Figure 5 shows a sentence that has been annotated for the terrorism domain: "A POLICEMAN" has been tagged as an injury victim, the "URBAN GUERRILLAS" have been tagged as the perpetrators of the attack, "THE GUARDS" have been tagged as victims, and "SAN SALVADOR" has been tagged as the location of the attack.

For the experiments described in this paper, we used the MUC-4 and MUC-5 answer keys as input to AutoSlog instead of an annotated corpus because they were available and contain the information that AutoSlog needs. However, they also contain information that AutoSlog does not need. In fact, AutoSlog did not use a lot of the information contained in the templates. An annotated corpus is sufficient for AutoSlog and much easier to generate for a new application. Throughout this paper, we will refer to AutoSlog's input as a "training corpus", which could be an annotated corpus or a set of texts and associated answer keys.

STEP 2: IDENTIFY THE SYNTACTIC ROLE OF THE TARGETED INFORMATION

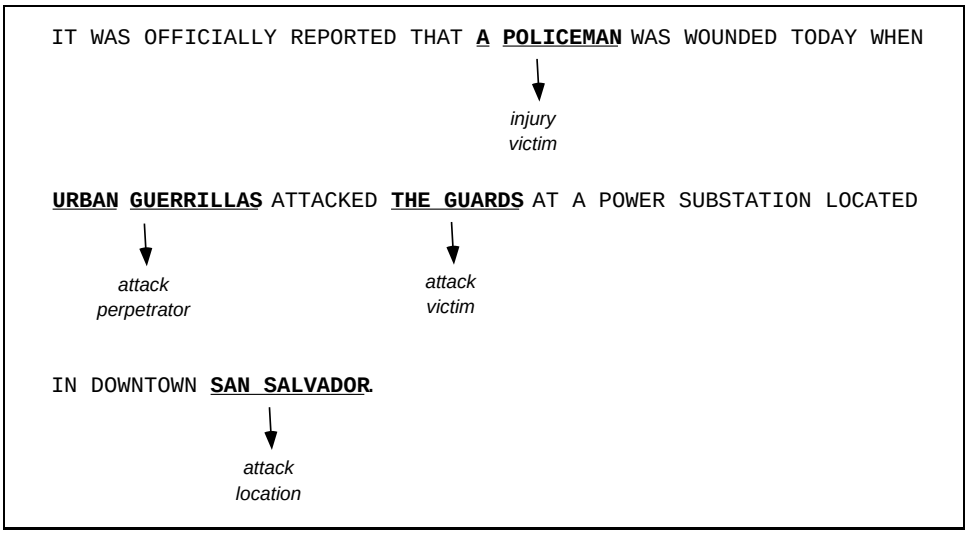

Fig. 5. Example text annotations for AutoSlog

For each targeted noun phrase in the training corpus, AutoSlog identifies the sentence from which it should be extracted. Given an annotated corpus, AutoSlog can just grab the sentence in which the noun phrase was tagged. Given a corpus of texts and answer keys, AutoSlog must map the targeted information back to the original source text. In this case, AutoSlog makes the assumption that the first sentence containing the noun phrase is the one from which it should have been extracted. This assumption is based on the fact that the MUC corpora consist mainly of newswire articles. Stylistically, news articles have the property that the most important information is usually reported first. Secondary information and details are usually reported later. For example, an article about the assassination of a mayor probably mentions that the mayor was assassinated before it provides details about his political career and family.

Given a targeted noun phrase and the sentence from which it should be extracted, AutoSlog passes the sentence to CIRCUS for syntactic analysis. CIRCUS' syntactic analyzer generally assigns each noun phrase to one of three syntactic categories: subject, direct object, or prepositional phrase. AutoSlog then identifies the syntactic category of the noun phrase.

STEP 3: IDENTIFY A TRIGGER WORD FOR A CONCEPT NODE

Given the syntactic category of the targeted noun phrase, a small set of heuristics is used to identify a trigger word. Intuitively, the trigger word should be the word that determines the conceptual role of the noun phrase (e.g., whether someone is a victim or perpetrator). For example, it is impossible to look at a name such as "John Smith" and determine whether John Smith is a victim or a perpetrator. His role is defined by the surrounding context. The sentence "John Smith was killed" identifies John as a victim, and the sentence "John Smith killed a man" identifies John as a perpetrator. In both cases, the verb

"killed" determines the conceptual role that John played in the event. In general, we will refer to this word as a "conceptual anchor point." With respect to CIRCUS, a conceptual anchor point is a trigger word for a concept node.

Figure 6 shows the set of thirteen *conceptual anchor point heuristics* used by AutoSlog. The heuristics do two things: (a) they identify the conceptual anchor point (trigger word) for a concept node, and (b) they identify the surrounding context that the concept node needs to recognize. The first column of Figure 6 shows the general patterns recognized by the heuristics, where the bracketed item identifies the syntactic category of the targeted noun phrase (subject, direct object, or prepositional phrase). The second column shows an example of how each pattern might be instantiated by AutoSlog; the underlined word is the trigger word and the bracketed item shows the conceptual role assigned to the extracted information.

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 prep <noun-phrase>	bomb against <target>
active-verb prep <noun-phrase>	killed with <instrument>
passive-verb prep <noun-phrase>	was aimed at <target>

Fig. 6. AutoSlog heuristics and examples from the terrorism domain

The heuristics fall into three sets based on the syntactic category of the targeted noun phrase. The first set of heuristics applies when the noun phrase is the subject of a clause. In this case, the verb is used as the trigger word because the verb determines the conceptual role of the subject. Several different verb forms are recognized. If the verb is in a passive construction, then the pattern must recognize passive verb forms. If the verb is in an active construction, then the pattern must recognize active verb forms. If an active verb is followed by an infinitive, then the infinitive is included in the pattern. For example, given the sentence "he intended to kill the president", the pattern "<perpetrator> intended to kill" is more informative than just "<perpetrator> intended." A special pattern handles the case where the verb is an auxiliary verb (i.e., "to be" or "to have"). These verbs do not convey much semantic information on

their own, so the head noun of the direct object is included in the pattern. For example, given the sentence "John was the fifth fatality", the pattern "<victim> was fatality" is more informative than "<victim> was."

The second set of heuristics applies when the targeted noun phrase is the direct object of a verb. In this case, the verb is also used as the trigger word because the verb determines the conceptual role of the object. The verb is almost always in an active or infinitive construction.[2] There are a few special cases. If the verb is followed by an infinitive then the infinitive is included in the pattern. If the verb is an auxiliary verb, then the head noun of the subject is included in the pattern. And one heuristic recognizes gerunds that take direct objects. For example, given the sentence "The FMLN has been accused of killing peasants." and the targeted noun phrase "peasants", a concept node would be generated for the pattern "killing <victim>," which is activated by the gerund form of "killing."

The third set of heuristics applies when the targeted noun phrase is in a prepositional phrase. In this case, a prepositional phrase attachment algorithm attaches the prepositional phrase to a noun or verb preceding it. The noun or verb chosen as the attachment point is combined with the preposition to form the pattern for a concept node.[3] In most cases, the heuristics are mutually exclusive so only one will fire for a given noun phrase. In the few cases where multiple rules apply, the longest pattern is selected.

2.3 Examples from the Terrorism Domain

To illustrate how AutoSlog works, we will show a few examples of concept node definitions created by AutoSlog for the terrorism domain. Figure 7 shows a sentence about a bombing incident. The noun phrase "public buildings" has been tagged as the target of the bombing. CIRCUS analyzes this sentence and identifies the "public buildings" as the subject of the first clause. The conceptual anchor point heuristics recognize the **<subject> passive-verb** pattern and produce a concept node to recognize expressions such as "<target> was bombed." This concept node is activated by passive forms of the verb "bombed", and extracts its subject as the target of a bombing. This concept

[2] In principle, passive verbs should not have direct objects but we included this pattern because CIRCUS occasionally confused active and passive verb forms.

[3] The pp-attachment algorithm used by AutoSlog is separate from CIRCUS. If the preposition is "of", "against", or "on", then the algorithm attaches the prepositional phrase to the most recent constituent. Otherwise, the algorithm attaches the prepositional phrase to the most recent verb or noun phrase but skips over intervening prepositional phrases. This algorithm makes a lot of mistakes and was intended only as a simple attempt to handle pp-attachment.

node represents a useful pattern for the terrorism domain because it is likely to appear in many stories about bombings.

Sentence: In La Oroya, Junin department, in the central Peruvian mountain range, public buildings were bombed and a car-bomb was detonated.

CONCEPT NODE

Name:	target-subject-passive-verb-bombed
Trigger:	bombed
Variable Slots:	(target (*SUBJECT* 1))
Constraints:	(class PHYS-TARGET *SUBJECT*)
Constant Slots:	(type bombing)
Enabling Conditions:	(passive)

Fig. 7. Concept node definition for "<target> was bombed"

Figure 8 shows an example of a concept node that recognizes a more complicated expression. Given the noun phrase "guerrillas" tagged as perpetrators, CIRCUS identifies the "guerrillas" as the subject of the first clause. The conceptual anchor point heuristics recognize the pattern **<subject> verb infinitive** and produce a concept node that is activated by the expression "threatened to murder." This concept node is triggered by the word "murder" but has enabling conditions that require it to be preceded by the words "threatened to." When the concept node is activated, it extracts the subject as a perpetrator. This concept node is also useful for the terrorism domain because it is likely to appear in many texts that describe death threats.

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. 8. Concept node definition for "<perpetrator> threatened to murder"

However, AutoSlog does not always generate concept nodes that represent useful expressions. Figure 9 shows a concept node produced by AutoSlog that recognizes expressions of the form "took <Y>." AutoSlog identified the targeted noun phrase, "Gilberto Molasco", as the direct object of the first clause and constructed a concept node that is triggered by the verb "took" and extracts

its direct object as a kidnapping victim. This concept node works correctly in the sentence it was given; Gilberto Molasco was indeed a kidnapping victim. But the expression "took <Y>" does not always apply to kidnappings. The word "took" commonly appears in many contexts. For example, one can take a friend to the movies or take a child to school.

Sentence: They took 2-year-old Gilberto Molasco, son of Patricio Rodriguez, and 17-year-old Andres Argueta, son of Emimesto Argueta.

CONCEPT NODE

Name:	victim-active-verb-dobj-took
Trigger:	took
Variable Slots:	(victim (*DIRECT-OBJECT* 1))
Constraints:	(class VICTIM *DIRECT-OBJECT*)
Constant Slots:	(type kidnapping)
Enabling Conditions:	(active)

Fig. 9. Concept node definition for "took <victim>"

Figure 10 shows another example of a concept node that represents an unreliable pattern. AutoSlog found the targeted noun phrase "machineguns" in a prepositional phrase and the pp-attachment algorithm incorrectly attached it to the noun "priests." The resulting concept node is activated by the pattern "priests with <X>" and extracts X as a weapon. This pattern is not likely to be reliable because priests aren't usually associated with weapons. If the pp-attachment algorithm had correctly attached the machineguns to the word "killing", then AutoSlog would have produced a better concept node that recognized the pattern "killing with <weapon>."

Sentence: Ambassador William Walker, if you still have any shame, tell the world and answer this question: if the armed forces general staff did not kill the jesuit priests, how could the murderers – as this international dispatch says – remain in the residence for 1 hour after the heavy shooting, after killing the priests with machineguns in tripods, as the cable says?

CONCEPT NODE

Name:	instrument-pp-noun-priests-with
Trigger:	priests
Variable Slots:	(instrument (*PREP-PHRASE* (pp-check 'with)))
Constraints:	(class WEAPON *PREP-PHRASE*)
Constant Slots:	(type weapon)
Enabling Conditions:	(noun-triggered)

Fig. 10. Concept node definition for "priests with <instrument>"

2.4 Results for the Terrorism Domain

To evaluate AutoSlog, we created a concept node dictionary for the MUC-4 terrorism domain using AutoSlog and compared it with the hand-crafted dictionary used in MUC-4.[4] We used 772 relevant texts from the MUC-4 development corpus and their answer keys as the training corpus. The targeted noun phrases came from six of the MUC-4 template slots that corresponded to human targets, physical targets, perpetrators, and weapons. These six information types, shown in Figure 11, were selected because the answer keys contained strings that could be easily mapped back to the source text.

Information Type	**Example**
human target description	"a security guard"
human target name	"Ricardo Castellar"
instrument id	"car-bomb"
perpetrator individual	"a group of subversives"
perpetrator organization	"the FMLN"
physical target id	"car dealership"

Fig. 11. Targeted information for the terrorism domain

The 772 texts contained 4780 tagged noun phrases of these six types, which were given to AutoSlog as input along with the original source texts.[5] In response to these 4780 noun phrases, AutoSlog generated 1237 unique concept node definitions. AutoSlog does not necessarily generate a concept node for every input. For example, sometimes none of the heuristics apply or CIRCUS produces a faulty sentence analysis. Also, AutoSlog does not generate duplicate definitions. For example, many texts contain expressions of the form "X was kidnapped" so AutoSlog will propose this pattern many times in response to different inputs. AutoSlog keeps track of the number of times each concept node is proposed, but will not generate the same definition twice. Figure 12 shows the patterns of the fifteen concept nodes that were proposed most frequently by AutoSlog. For example, AutoSlog proposed a concept node to recognize the pattern "<victim> was kidnapped" 46 times.

[4] In fact, this was a slightly improved version of the hand-crafted dictionary used in MUC-3. We augmented the hand-crafted dictionary with 76 concept nodes created by AutoSlog before the final MUC-4 evaluation, which improved the performance of the UMass/MUC-4 system by filling gaps in its coverage. Without these additional concept nodes, the AutoSlog dictionary would likely have shown even better performance relative to the MUC-4 dictionary.

[5] Many of the template slots contained several possible references to the same object ("disjuncts"), any one of which was a legitimate answer. In this case, AutoSlog identified the first sentence that contained any of the references.

Linguistic Pattern	Number of Times Proposed
<victim> was killed	121
murder of <victim>	111
assassination of <victim>	95
<victim> was wounded	50
<victim> was kidnapped	46
<weapon> exploded	43
killed <victim>	42
death of <victim>	40
murdered <victim>	36
<victim> died	35
<victim> was murdered	34
<perpetrator> attacked	32
<victim> was injured	29
<victim> was assassinated	29
kidnapped <victim>	29

Fig. 12. Frequently proposed patterns for terrorism

As we mentioned in the previous section, AutoSlog generates many useful concept nodes but it also generates many unreliable concept nodes. Therefore we put a human in the loop to weed out the unreliable definitions. We developed a simple user interface that displays the pattern associated with each concept node to a user and asks whether the concept node should be accepted or rejected. The concept nodes rejected by the user are thrown away, and the concept nodes accepted by the user are retained for the final dictionary.

The process of manually filtering the dictionary is very fast and does not require any knowledge of CIRCUS or natural language processing. For this experiment, a second-year graduate student with some knowledge of CIRCUS and NLP manually filtered the terrorism dictionary. It took him 5 hours to review all 1237 concept node definitions and he accepted 450 of them for the final dictionary. Figure 13 shows the distribution by types. The first column shows the number of concept nodes proposed by AutoSlog, and the second column shows the number of concept node accepted by the user (e.g., the user accepted 34 of the 191 human target description concept nodes). Overall, 36% of the concept nodes proposed by AutoSlog were accepted for the final dictionary.

Finally, we compared the dictionary created by AutoSlog with the hand-crafted dictionary. We took the official UMass/MUC-4 system, removed the hand-crafted dictionary, and replaced it with the AutoSlog dictionary. The two information extraction systems were therefore identical except that they used different concept node dictionaries.[6] We then scored the official MUC-4

[6] We also added four manually constructed concept node definitions to the AutoSlog dictionary because they were important for discourse analysis. These special concept

CN Type	#CNs Proposed	#CNs Kept
human target description	191	34
human target name	169	51
instrument id	129	93
perpetrator individual	303	102
perpetrator organization	165	31
physical target id	280	139
TOTAL	1237	450

Fig. 13. Acceptance rates for the terrorism dictionary

system (with the hand-crafted dictionary) and the AutoSlog version using the MUC-4 scoring program [27]. The results appear in Figure 14.

System/Test Set	Recall	Precision	F-measure
MUC-4/TST3	46	56	50.51
AutoSlog/TST3	43	56	48.65
MUC-4/TST4	44	40	41.90
AutoSlog/TST4	39	45	41.79

Fig. 14. Comparative results

The MUC-4 scoring program generated recall and precision scores as well as an f-measure score. Recall measures the percentage of correct information that was extracted by the system; intuitively, how much of the desired information the system found. Precision measures the percentage of information that the system extracted which was actually correct; intuitively, how often the system was correct when it extracted something. The f-measure combines both recall and precision, in this case with equal weighting.

Both systems were evaluated on two blind test sets of 100 texts each, TST3 and TST4. Figure 14 shows that the AutoSlog dictionary achieved performance comparable to the hand-crafted dictionary. On TST3, the AutoSlog dictionary achieved 96.3% of the performance of the hand-crafted dictionary, comparing f-measures. On TST4, the f-measures were almost indistinguishable, with the AutoSlog dictionary achieving 99.7% of the performance of the hand-crafted dictionary. The hand-crafted dictionary achieved higher recall than the AutoSlog dictionary on TST4, but the AutoSlog dictionary achieved higher precision.

Overall, the dictionary created by AutoSlog achieved 98% of the performance of a dictionary that was built manually, with substantially less time required for knowledge engineering. Although the hand-crafted dictionary required approximately 1500 person-hours to build, the AutoSlog dictionary required only

nodes were not used to extract information, but only to identify textual cues for discourse purposes.

5 person-hours for filtering plus the time required to generate the training corpus.[7] Furthermore, building a concept node dictionary by hand requires experienced system developers, but no experience is required to filter dictionaries produced by AutoSlog. We will present empirical results to support this claim in Section 4.

3 Moving AutoSlog to New Domains

The previous experiment showed that a concept node dictionary produced by AutoSlog performed well in the terrorism domain. However, we wanted to know whether AutoSlog could produce effective dictionaries for other domains as well, so we generated concept node dictionaries for two additional domains: a business-oriented domain of joint venture activities, and a technical domain, microelectronics. We chose these domains because they were the focus of the MUC-5 evaluation and we had access to large training corpora of texts and answer keys. The domains also represent very different topics, and were therefore a good testbed for evaluating the generality of AutoSlog.

Because we participated in MUC-5 as part of the NLP group at the University of Massachusetts, the dictionaries produced by AutoSlog were used by the UMass/MUC-5 system. AutoSlog's heuristics are domain-independent so porting AutoSlog to the new domains was easy. However, we needed AutoSlog to generate the best dictionaries that it possibly could. Therefore, our purposes were twofold: (1) to determine whether the domain-independent heuristics could produce effective concept nodes for different domains, and (2) to determine whether the heuristics (or possibly the whole approach) needed to be modified. We were fully prepared to make significant changes to AutoSlog if we felt that the original heuristics were not adequate. In the next section, we discuss improvements to AutoSlog for these new domains.

3.1 Improvements and Modifications to AutoSlog

Our strategy was to apply AutoSlog to the new domains, review the resulting concept node definitions, and make changes to AutoSlog as needed. In the end, we were pleasantly surprised to find that the original set of heuris-

[7] The answer keys used in this experiment contained a lot of information that AutoSlog did not use, so we cannot estimate the time required to generate an appropriate training corpus based on the time it took to generate the answer keys. However, preliminary experiments showed that a user can annotate 160 texts in about 8 hours.

tics performed well and required few modifications. However, we added a few capabilities to AutoSlog to improve its performance.

We made only three changes to the heuristics. Two of these changes were minor, but one was more significant. First, the **passive-verb <direct-object>** pattern was dropped. This heuristic was used in the terrorism system only because early versions of CIRCUS had trouble distinguishing active and passive verb forms. In principle, this heuristic should never have fired unless CIRCUS made a mistake. Second, a new pattern was added: **infinitive preposition <noun-phrase>**. This heuristic represents patterns such as "to collaborate on a project." We simply hadn't seen this pattern in the terrorism domain, probably because terrorist events are usually reported in the past tense. Joint venture activities, however, are often reported in the future tense.

The third, more significant change was another new pattern: **<subject> verb direct-object**, which represents expressions such as "Toyota and Nissan formed a joint venture." This pattern reflects an important difference between the language typically used to describe terrorist events and the language used to describe joint ventures. Verbs usually carry the semantics associated with terrorist events. For example, the words "bombed, "murdered", and "kidnapped", commonly describe terrorist events. However, nouns typically carry the semantics associated with joint ventures while the verbs are relatively weak. For example, common expressions are: "X and Y formed a joint venture", "X agreed to a tie-up with Y", or "X signed an agreement with Y." The verbs (formed, agreed, and signed) are not specific to joint ventures; the nouns (venture, tie-up, agreement) are the words most strongly associated with joint ventures.

The original **<subject> active-verb** heuristic would have proposed concept nodes to recognize expressions such as "X formed", "X agreed", and "X signed." These patterns are too general and will extract a lot of irrelevant information. Therefore, we added the new **<subject> verb direct-object** heuristic to include the direct object as part of the pattern. If a direct object is present, then this heuristic takes precedence over the original one and a concept node is generated using both the verb and the head noun of its direct object. If a direct object is not present, then AutoSlog falls back on the original heuristic. The new pattern produced many useful concept nodes for the joint ventures domain, including expressions such as "X formed venture", "X completed acquisition", and "X signed agreement." The modified set of AutoSlog heuristics appears in Figure 15.

A few other modifications were made as well. In the joint ventures domain, particles play an important role in many expressions, such as "set up venture", "linked up with", and "carrying out study." The heuristics that include verbs were modified so that AutoSlog searches for a particle immediately fol-

Linguistic Pattern	Example
<subject> passive-verb	<entity> was formed
<subject> active-verb	<entity> linked
<subject> verb direct-object	<entity> completed acquisition
<subject> verb infinitive	<entity> agreed to form
<subject> auxiliary noun	<entity> is conglomerate
active-verb <direct-object>	acquire <entity>
infinitive <direct-object>	to acquire <entity>
verb infinitive <direct-object>	agreed to establish <entity>
gerund <direct-object>	producing <product-service>
noun auxiliary <direct-object>	partner is <entity>
noun prep <noun-phrase>	partnership between <entity>
active-verb prep <noun-phrase>	buy into <entity>
passive-verb prep <noun-phrase>	was signed between <entity>
infinitive prep <noun-phrase>	to collaborate on <product-service>

Fig. 15. AutoSlog heuristics and examples from the joint ventures domain

lowing the verb. For example, given the sentence "company X was set up ... ", the **<subject> passive-verb** heuristic fires and finds the particle "up" following the verb "set." The resulting concept node represents the pattern "<entity> was set up", which is more appropriate than just "<entity> was set." Particle recognition would have been useful in the terrorism domain as well for expressions such as "blew up", "blown up", and "carried out", but the UMass/MUC-4 system used a hand-crafted phrasal lexicon to identify these expressions. In retrospect, AutoSlog could have automatically created concept nodes to recognize many of the expressions that were manually encoded in the terrorism phrasal lexicon.

Another improvement to AutoSlog involved objects with computable values. For example, ownership percentages and monetary values are prevalent in the joint ventures domain. The original version of AutoSlog produced concept nodes that recognized overly specific patterns, such as "<entity> controls 51%", and "<entity> invested $50000000." To address this problem, we modified AutoSlog so that concept nodes can be triggered by general types of objects (e.g., percentages and monetary figures). For example, given the sentence "IBM controls 51%...", the **<subject> verb direct-object** heuristic fires and recognizes that the head noun of the direct object is a percentage. AutoSlog then proposes a concept node that is activated by all expressions of the form "<entity> controls PERCENTAGE." The UMass/MUC-4 system contained specialist functions to recognize percentages and monetary values, which were used to identify these objects.

For the sake of completeness, we will briefly mention a few other changes. We replaced the original pp-attachment algorithm with a frequency-based pp-

attachment algorithm (see [31] for details). We divided the heuristics involving auxiliary verbs (**<subject> auxiliary noun** and **noun auxiliary <direct-object>**) into separate heuristics that distinguish between the verbs "to be" and "to have." And we modified AutoSlog to skip over clauses that contain communication verbs, such as "said", "reported" and "announced", since they merely indicate that something is being reported. Finally, we added a morphology component that automatically generates morphological variants of proposed patterns. For example, if AutoSlog generates a concept node triggered by a singular noun then a new concept node is generated dynamically for the same pattern with the plural noun. All morphological variants were presented to the user for manual filtering. [8]

These changes were all general improvements that would have applied to the terrorism domain as well. The only modification made to AutoSlog that appears to be domain-specific is the addition of the **<subject> verb direct-object** pattern. In the next two sections, we describe the dictionaries generated for the joint ventures and microelectronics domains.

3.2 Results for the Joint Ventures Domain

The joint ventures information extraction task revolves around cooperative agreements between multiple partners, usually to jointly produce a product or service. Figure 16 shows the eight types of information for which concept nodes were generated. The most important information corresponds to the names of the entities involved in the joint venture; relevant entities can be companies, people, or governments. Other relevant information includes facilities, products, services, and people associated with a joint venture, the ownership percentage of entities, and several monetary values.

These types of information cannot be identified without context! Many company names can be recognized simply by looking for abbreviations such as Corp. or Inc.. But we only want to extract the names of companies that are involved in a joint venture. Therefore, simply looking for patterns such as "X Corp." or "X Inc." will likely produce many false hits by extracting companies that have nothing to do with a joint venture. Similarly, monetary figures and percentages can be easily recognized but we only want to extract them if they are associated with a joint venture.

[8] This component was not necessary for the terrorism domain because the UMass/MUC-4 system contained a morphological analyzer so each concept node was automatically triggered by all morphological variants. The UMass/MUC-5 system did not contain a morphological analyzer, however, so separate concept nodes had to be created for each variant.

Information Type	Example
entity name	"Toyota Motor Corp."
facility name	"Beijing jeep plant"
ownership percent	"51%"
ownership total capitalization	"$46,000,000"
person name	"Paul Phillips"
product/service	"V2500 jet engine"
revenue rate	"$80,000,000 per year"
revenue total	"$80,000,000"

Fig. 16. Targeted information for the joint ventures domain

Figure 17 shows a concept node generated by AutoSlog for the joint ventures domain. Given the targeted noun phrase "Berliner Bank", AutoSlog identified the bank as the subject of the first clause. The new **<subject> verb direct-object** heuristic kicked in and produced a concept node that is activated by the expression "<X> formed venture" and extracts X as a joint venture entity (i.e., partner). This concept node represents a reliable pattern associated with joint ventures.

Sentence: Berliner Bank last year formed a joint venture with KFTCIC to channel investment into medium-sized German companies.

CONCEPT NODE

Name:	jv-entity-subject-verb-and-dobj-formed-venture
Trigger:	venture
Variable Slots:	(name (*SUBJECT* 1))
Constraints:	(class JV-ENTITY *SUBJECT*)
Constant Slots:	(type jv-entity subtype company relationship jv-parent)
Enabling Conditions:	(dobj-preceded-by-verb 'formed 'venture)

Fig. 17. Concept node definition for "<entity> formed venture"

As in the terrorism domain, not all of the concept nodes generated by AutoSlog were useful. Figure 18 shows a bizarre concept node produced by AutoSlog. The targeted noun phrase, ICI, was identified as the subject of the verb "thrown." The new **<subject> verb direct-object** heuristic kicked in and generated a concept node that recognizes the pattern "<entity> thrown hat." The metaphorical expression "thrown its hat into the ring" is not usually associated with joint ventures, so this concept node was rejected.

As input, AutoSlog was given 924 relevant texts from the MUC-5 joint ventures corpus that contained 10,684 targeted noun phrases. The overwhelming majority represented entities (mostly companies) and products or services associated with them. Figure 19 shows statistics for the joint ventures dictionary. The first column shows the number of targeted noun phrases. The second column shows the number of concept nodes generated by AutoSlog. The third

Sentence: In addition to Japanese, Taiwanese and South Korean firms, ICI has thrown its hat into the ring with 350000 ton ayear PTA plants in Taiwan and Thailand.

CONCEPT NODE

Name:	jv-entity-subject-verb-and-dobj-thrown-hat
Trigger:	hat
Variable Slots:	(entity (*SUBJECT* 1))
Constraints:	(class JV-ENTITY *SUBJECT*)
Constant Slots:	(type jv-entity subtype company relationship jv-parent)
Enabling Conditions:	(dobj-preceded-by-verb 'thrown 'hat)

Fig. 18. Concept node definition for "<entity> thrown hat"

column shows the number of concept nodes that were accepted by the user. And the fourth column shows the total number of concept nodes accepted for the final dictionary, including the ones generated by the morphology module. When a concept node was accepted, morphological variants of the pattern were generated dynamically and the user was asked whether any of the variants should be accepted as well. For example, if the user accepted the pattern "X formed venture", then new concept nodes were created for the patterns "X form venture", "X forms venture", "X forming venture", and "X formed ventures." On average, 1.7 morphological variants were accepted for each original concept node.

CN Type	#NPs	#CNs Proposed	#CNs Kept	#CNs Kept w/Morph. Variants
entity	4689	1562	527	1570
facility	97	80	20	38
ownership percent	814	174	90	184
ownership total capitalization	139	25	14	16
person	554	243	119	355
product/service	4296	1034	138	273
revenue rate	50	19	14	22
revenue total	45	30	22	57
TOTAL	10,684	3167	944	2515

Fig. 19. AutoSlog dictionary statistics for joint ventures

The human-in-the-loop took 20 hours to review the 3167 concept nodes proposed by AutoSlog (the human-in-the-loop for this experiment was the author). This is substantially more time than it took to review the terrorism definitions (5 hours). The increased time is due to two factors. First, AutoSlog proposed 2.6 times as many definitions for the joint ventures domain (3167) as for the terrorism domain (1237), primarily because AutoSlog received 2.2 times as many noun phrases for joint ventures (10,684) as for terrorism (4780).

Second, a lot of the increased filtering time is due to the overhead associated with the morphology module, which substantially increased the number of definitions displayed to the user. Consequently, the filtering processes for the joint ventures and terrorism dictionaries were not directly comparable.

Evaluating the joint ventures dictionary is difficult because we did not have a hand-crafted dictionary with which to compare it, and building one by hand is expensive. Alternatively, we could compare the UMass/MUC-5 results with the UMass/MUC-4 results and infer that the new dictionary performs well if we obtain similar results. However, this is not a valid comparison because the MUC-4 and MUC-5 systems were almost completely different. The UMass/MUC-5 system used a different part-of-speech tagger, noun phrase bracketer, word sense disambiguation module, and discourse analyzer. The only common component was the sentence analyzer, CIRCUS.

The UMass/MUC-5 system achieved scores of 26% recall and 54% precision (f-measure = 35.18) for the joint ventures domain. Therefore we can infer a lower bound on the performance of the AutoSlog dictionary: it was able to extract at least 26% of the desired information. [9] However, we believe that the dictionary actually performed much better than these numbers would suggest. In the next section, we describe a small experiment in which we manually inspected 25 random texts and found that CIRCUS actually achieved 68% recall on those texts.

Linguistic Pattern	**Times Proposed**
venture with <entity>	230
agreement with <entity>	54
venture between <entity>	51
<entity> formed venture	45
was owned by <entity>	39
<entity> agreed	38
<entity> set up venture	37
<entity> was capitalized	35
subsidiary of <entity>	34
<entity> signed agreement	34
unit of <entity>	34
PERCENTAGE by <entity>	29
<entity> agreed to form	27

Fig. 20. Frequently proposed joint venture patterns

AutoSlog clearly created many patterns that were appropriate for the joint ventures domain and CIRCUS appeared to be doing a good job of extracting most of the relevant information. Figure 20 shows the concept nodes most

[9] This should be interpreted with respect to the current state-of-the-art in information extraction. The best information extraction systems at MUC-4 obtained roughly 50-60% recall using hand-crafted dictionaries.

frequently proposed by AutoSlog. As might be expected, many frequent patterns include the word "venture", "agreement", or "agreed." Other relevant patterns represent expressions having to do with ownership, capitalization, or percentages. As Figure 19 indicated, a user ultimately accepted 944 of the original concept nodes as being good extraction patterns, plus an additional 1571 morphological variants of those patterns. Therefore a human judged that 944 of AutoSlog's definitions were desirable extraction patterns, plus over 1500 morphological variants. In the end, the filtered joint ventures dictionary was substantially bigger than the terrorism dictionary and presumably provided better coverage as a result.

3.3 Results for the Microelectronics Domain

The microelectronics information extraction task was concerned with information about four microelectronics processes: layering, lithography, etching, and packaging. To be relevant, a specific company or research group had to be associated with one of these process types. Figure 21 shows the twelve information types for which concept nodes were generated.

Information Type	**Example**
bonding type	LASER_BONDING
device function	MICROPROCESSOR
device size	64 MBIT
device speed	70 MHZ
entity name	"Material Research Corp."
equipment name	"Precision 8000"
equipment type	CVD_SYSTEM
film type	SILICON_DIOXIDE
granularity size	LINE WIDTH 0.25MI
material type	CERAMIC
pin count	408
process type	CHEMICAL VAPOR DEPOSITION

Fig. 21. Targeted information for microelectronics

The microelectronics task was fundamentally different from the terrorism and joint ventures tasks because the information to be extracted was delimited in advance. The MUC-5 guidelines contained a finite list of the legitimate values for 10 of the 12 information types. For example, the guidelines listed all of the legitimate bonding types. In a few cases, the guidelines listed units (e.g., MBIT and MHZ) for which numbers had to be extracted (e.g., device size and speed). Words or phrases that did not match one of the predefined values did not have to be extracted. In contrast, arbitrary values needed to be extracted for the terrorism and joint ventures domains, so the set of legitimate values

could not be predetermined. Only two information types could take arbitrary strings in the microelectronics domain: entity names and equipment names.

Figure 22 shows a good concept node produced by AutoSlog to extract entities. "Fujitsu Laboratories" was given to AutoSlog as input and CIRCUS identified it as the subject of the first clause. The **<subject> verb direct-object** heuristic fired and produced a concept node that recognizes the pattern "<entity> developed technology." This pattern is not specific to microelectronics and could extract companies that develop other types of technology. But this pattern will appear in many texts describing microelectronics technology, so it should be retained or a lot of relevant information will be missed.

Sentence: Fujitsu Laboratories has developed a technology to selectively form a two-dimensional electron gas layer on top of an electron donor layer.

CONCEPT NODE

Name:	me-entity-subject-verb-and-dobj-developed-technology
Trigger:	technology
Variable Slots:	(name (*SUBJECT* 1))
Constraints:	(class ME-ENTITY *SUBJECT*)
Constant Slots:	(type me-entity subtype company relationship developer)
Enabling Conditions:	(dobj-preceded-by-verb 'developed 'technology)

Fig. 22. Concept node definition for "<entity> developed technology"

Figure 23 shows a concept node produced by AutoSlog to extract microelectronics processes, such as layering and lithography. In the given sentence, the targeted noun phrase is "MBE" (molecular beam epitaxy). AutoSlog identified "MBE" as the direct object of the verb "using" and created a concept node for the pattern "using <X>." Although this pattern extracts a relevant process in this particular sentence, "using" is a general verb that appears in a wide variety of contexts. There is a balance that must be maintained between generality and specificity. Overly general patterns will swamp the discourse analyzer with irrelevant information and merely shift the burden of identifying relevant information to later stages of processing. This concept node is therefore not particularly useful because it is likely to extract a lot more irrelevant than relevant information.

We applied AutoSlog to 787 relevant microelectronics texts from the MUC-5 corpus.[10] Figure 24 shows the ten concept nodes that were proposed most frequently by AutoSlog. The patterns are not as specific as those for the joint

[10] One of these texts was classified as relevant when we did these experiments but was reclassified as irrelevant by the MUC-5 organizers before the final evaluation. Therefore the MUC-5 microelectronics corpus officially contains 786 relevant texts.

Sentence: To form the layer, the laboratory developed a continuous process for growing crystals in an ultra-high vacuum environment using MBE, a method of selectively implanting impurities with an FIB (focused ion beam) method, and adopted a high-speed heat treating process.

CONCEPT NODE

Name:	me-process-type-dobj-verb-using
Trigger:	using
Variable Slots:	(name (*DIRECT-OBJECT* 1))
Constraints:	(class ME-PROCESS *DIRECT-OBJECT*)
Constant Slots:	(type me-process subtype layering)
Enabling Conditions:	(active)

Fig. 23. Concept node definition for "using <process>"

ventures domain, but most of them are likely to extract companies or equipment associated with microelectronics processes. However, AutoSlog did not produce many concept nodes that were useful for extracting the other 10 types of information (called the *set-fill* types). Most of the concept nodes represented patterns that were too general and would have extracted an overwhelming amount of irrelevant information. This is because the words and phrases associated with microelectronics are almost exclusively noun phrases that are unambiguous and self-contained. For example, microelectronics processes include "physical vapor deposition" and "chemical vapor deposition" (CVD), equipment types include "stepper systems" and "CVD systems", and device functions include "microprocessor."

Linguistic Pattern	Number of Times Proposed
agreement with <entity>	18
researchers at <entity>	17
order from <entity>	14
manager at <entity>	14
includes <equipment-name>	13
<entity> developed technology	12
was developed by <entity>	12
order for <equipment-name>	11
introduced <equipment-name>	11
include <entity>	10

Fig. 24. Frequently proposed patterns for microelectronics

As we noted earlier, information associated with terrorism and joint ventures cannot be identified without context. It is not possible to look solely at a person's name and determine whether that person is a perpetrator or victim. Similarly, it is not possible to look only at a company's name and determine whether it is involved in a joint venture. Verbs (e.g., "was killed"), verb phrases (e.g., "formed venture"), and verb nominalizations (e.g., "assassination of") are essential for identifying the conceptual roles of these objects. However, it

is possible to look for specific microelectronics terms independent of context. The phrase "chemical vapor deposition" means essentially the same thing in almost any context. Furthermore, the set of technical terms specific to microelectronics is relatively small and finite (essentially a closed class). In contrast, the sets of potential perpetrators and joint venture companies are infinitely large. As a result, contextual patterns are essential for extracting most terrorism and joint ventures information but keywords and phrases are sufficient for recognizing microelectronics terms.

Figure 25 shows the number of concept nodes proposed by AutoSlog for each information type, the number of concept nodes accepted during manual filtering, and the total number of concept nodes in the final dictionary, including those generated by the morphology component. As Figure 25 shows, we did not filter the set-fill concept nodes.[11] Instead, we added a keyword recognizer to extract the microelectronics terminology. The keyword recognizer was combined with the concept nodes to capture role relationships associated with the microelectronics terms.[12] The set-fill concept nodes were all loaded into the system but information extracted by them was filtered by the keyword recognizer.

CN Type	**#CNs Proposed**	**#CNs Kept**	**#CNs Kept with Morph. Variants**
entity name	971	451	1445
equipment name	249	96	209
set-fill type	1732	1728	2566
TOTAL	2952	2275	4220

Fig. 25. AutoSlog dictionary statistics for microelectronics

The concept nodes were used by the discourse analyzer to identify relationships across items. For example, consider the sentence "A CVD system was developed by Motorola." Two concept nodes are triggered by the word "developed." First, a set-fill concept node is activated by the general pattern "X was developed" and extracts "a CVD system" as a product. The keyword recognizer identifies "CVD" as a microelectronics term so the information is considered to be relevant. Second, an entity concept node is activated by the pattern "was developed by Y" and extracts "Motorola" as a company name. The discourse analyzer can then link the CVD system to Motorola by virtue of the common verb "developed" that triggered both concept nodes. This approach shows how keyword recognition can be combined with concept nodes to handle both specialized terminology and conceptual role relationships.

[11] Only 1728 of the 1732 were kept because four definitions were discarded accidentally.

[12] The keyword recognizer was also used to identify relevant information independently from the concept nodes.

The UMass/MUC-5 system achieved scores of 31% recall and 39% precision (f-measure = 34.84) for the microelectronics domain. As before, we can infer a lower bound: CIRCUS was able to extract at least 31% of the desired information. However we believed that the performance of CIRCUS was much higher, so we conducted an experiment to assess its actual performance. Choosing 25 texts at random, we manually inspected the intermediate output and found that CIRCUS had extracted information with 68% recall and 54% precision. Obviously, much of the information was deleted or confounded by subsequent components (see [21] for more details). After discourse analysis, our official scores for these 25 texts were 32% recall and 45% precision, which is consistent with the overall results. If these texts were representative, then it appears that the MUC-5 system was able to achieve roughly 68% recall, which is actually higher than the recall reported by the UMass/MUC-4 system.

To conclude, we have shown that AutoSlog is a viable approach for automatically acquiring patterns for information extraction, and can produce effective extraction patterns for different domains. However, we learned a valuable lesson in applying the system to new domains. The nature of the domain is crucially important in determining what type of extraction patterns are necessary. In the terrorism domain, verbs often carry the semantics associated with an event so simple verb patterns were sufficient. In the joint ventures domain, nouns often carry the semantics associated with an event, so an additional heuristic was needed to pair nouns with verbs. And in the microelectronics domain, the technical jargon was most easily identified using keywords. The extraction patterns were useful, however, for identifying the roles associated with the technical information. We conclude that AutoSlog is most appropriate for recognizing role relationships between events and objects. The domain-independent heuristics used by AutoSlog are most well-suited for event-based domains.

4 Experiments with Novice Users

The previous experiments relied on a person to manually filter the dictionaries and discard unreliable definitions. From a practical perspective, it is important to know whether the filtering must be done by an expert (i.e., someone who is knowledgeable about natural language processing and CIRCUS in particular), or whether the filtering can be done by anyone knowledgeable about the domain. It is also important to have some idea of how much variation there is between dictionaries filtered by different people. So we set out to answer the following questions:

(1) Can people with little or no background in text processing create effective concept node dictionaries using AutoSlog?

(2) How much variation is there in the performance of dictionaries created by different people?

We addressed these questions by conducting two experiments with novice users (i.e., people who had little or no previous experience with CIRCUS). In the first experiment, we asked ten students in an introductory natural language processing course to filter the terrorism dictionary created by AutoSlog. In the second experiment, we asked two government analysts to filter the joint ventures dictionary created by AutoSlog.

4.1 An Experiment with Students in the Terrorism Domain

The first experiment involved ten students, including undergraduate and graduate students, in the introductory natural language processing course at the University of Massachusetts. Prior to this experiment, the students had received some exposure to CIRCUS in the form of 2 lectures, 1 paper, and 2 programming assignments. That had also been given 1 lecture and 1 paper on information extraction in the terrorism domain. So the students were not complete novices, in the sense that they had some knowledge about natural language processing and a little experience with an educational version of CIRCUS. But they had no experience with the UMass/MUC-4 system on which the dictionaries would be tested, except for one graduate student who we will refer to as Student X.

The students were given 1 hour of instruction on how to use the AutoSlog interface and were given two weeks to filter the terrorism dictionary produced by AutoSlog. We evaluated each dictionary by removing the hand-crafted dictionary from the UMass/MUC-4 system and replacing it with one of the student dictionaries. Then we ran the new system on the two blind test sets TST3 and TST4 (see Section 2.4), and scored the output using the MUC-4 scoring program [27].

Figure 26 shows the scores produced by the student dictionaries (these are the combined results for both TST3 and TST4). For the sake of comparison, we included the scores produced by the hand-crafted terrorism dictionary, denoted as MUC-4. Two of these data points are somewhat anomalous. Student X was a research assistant in the natural language processing lab and had some experience with the UMass/MUC-4 system, so his results should not be interpreted as those of a novice (although he was not one of the principal developers of the system). Student X's dictionary achieved the best performance, and was used in the experiments described in Section 2.4. The second anomalous data point is Student I. Student I was not a native English speaker and apparently did not understand the instructions given in class. We discovered that he did

System	Recall	Precision	F-measure
MUC-4	45	49	46.93
Student X	41	51	45.65
Student A	38	46	42.00
Student B	37	39	38.14
Student C	32	47	37.80
Student D	36	39	37.61
Student E	34	39	36.34
Student F	31	40	35.01
Student G	33	36	34.56
Student H	33	34	33.57
Student I	33	16	21.29

Fig. 26. Student dictionary scores on TST texts

not filter the dictionary at all, but kept every concept node proposed by AutoSlog! Therefore, the scores produced by Student I's dictionary represent an interesting baseline; they tell us how well the AutoSlog dictionary performs with no filtering at all.

If we disregard the data points associated with Student X and Student I, the range of scores is relatively small: the f-measures range from 33.57 to 42.00. There was a fair amount of variation in the performance of the dictionaries, but the scores were all within 9 points of one another so the differences were not extreme. The student dictionaries achieved 72-89% of the performance of the hand-crafted dictionary. Figure 27 shows the scatterplot for the recall and precision scores.

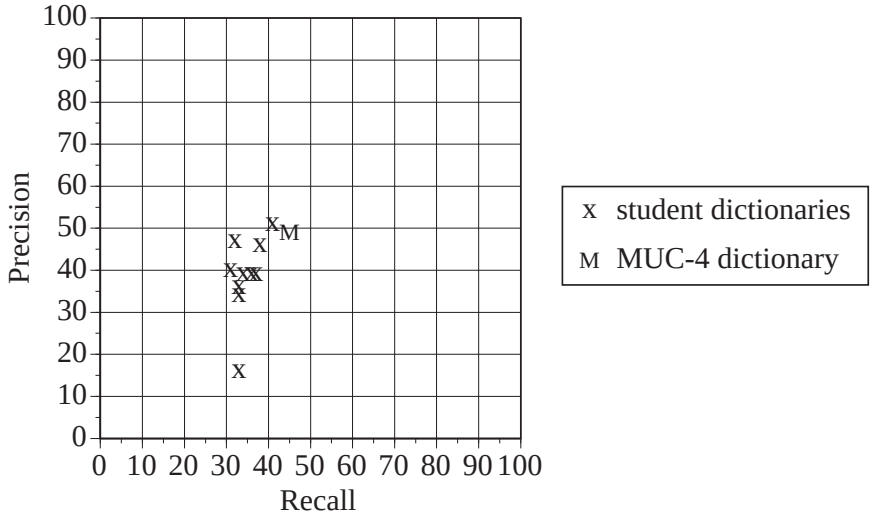

Fig. 27. Recall and precision scores for the student dictionaries

To put these numbers in perspective, consider how the scores of the student dictionaries compare with the scores of the MUC-4 participants. The best student dictionary (disregarding Student X) achieved an f-measure of 43.82

on TST3, which would have placed it fifth in the MUC-4 rankings (see [27]). Only four of the seventeen MUC-4 systems achieved higher scores. The student dictionary that obtained the lowest score on TST3 (35.57) would have ranked eighth in MUC-4. So all of the student dictionaries achieved TST3 scores better than half of the MUC-4 participants. On TST4, the highest-scoring student dictionary would have ranked seventh and the lowest-scoring dictionary would have ranked eleventh. We conclude that most of the concept node dictionaries produced by the students achieved scores that were better than or comparable to many of the MUC-4 systems.

Although the scores produced by the student dictionaries were not dramatically different, some dictionaries clearly performed better than others. Part of the reason is that the size of the dictionaries varied a lot. Figure 28 shows the number of concept node definitions accepted by each student, and the number of the definitions in the hand-crafted MUC-4 dictionary. Discounting Student I, who kept every definition, the dictionaries ranged in size from 304 to 645 definitions. Student F's dictionary contained over twice as many definitions as Student C's dictionary.

Dictionary	**# of Definitions**
Student C	304
MUC-4	389
Student A	390
Student H	399
Student B	422
Student X	450
Student E	478
Student D	567
Student G	619
Student F	645
Student I	1237

Fig. 28. Student dictionary sizes

Given the considerable variation in dictionary size, we tried to determine whether there was any correlation between dictionary size and performance. Figure 29 shows a scatterplot of the relationship between dictionary size and recall. There appears to be virtually no correlation. Some of the smallest dictionaries produced the highest recall, and both small and large dictionaries produced relatively low recall. Intuitively, one might assume that larger dictionaries should produce higher recall than smaller dictionaries. However, this is not necessarily the case. The information extraction task involves extracting relevant information and ignoring irrelevant information. Therefore, extracting irrelevant information does not increase recall. Furthermore, irrelevant information can complicate discourse analysis. When irrelevant information is given to the discourse analyzer, it often gets confused and may hallucinate

events and assign relevant information to imaginary events.

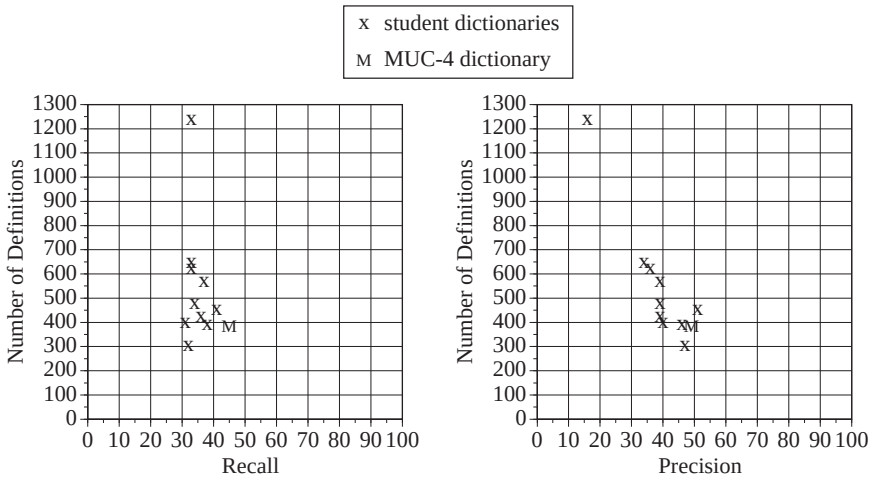

Fig. 29. Recall and precision vs. number of definitions

Figure 29 also shows the relationship between dictionary size and precision. Although there is not a perfect correlation, the graph suggests that smaller dictionaries tend to achieve higher precision than larger dictionaries. This makes sense if we assume that students who generated smaller dictionaries adopted a more conservative filtering strategy and retained only the most reliable definitions. Students who generated larger dictionaries probably adopted a more liberal strategy and retained definitions that may be useful in some cases but are prone to false hits.

The MUC-4 systems were also evaluated by how well their systems could distinguish stories that contained a relevant event from those that did not. This is a classification problem: each text had to be labeled as "relevant" or "irrelevant" to the domain. Roughly 53% of the texts in the MUC-4 corpus were relevant. Figure 30 shows the recall and precision scores computed by the MUC-4 scoring program for the student dictionaries on the classification task. There was less variation in the performance of the dictionaries on the classification task. Except for Student I, all of the dictionaries achieved at least 79% recall and 75% precision, and many achieved $\geq$ 85% recall with $\geq$ 80% precision. Almost all of the dictionaries performed nearly as well as the hand-crafted dictionary.

Despite the fact that the dictionaries varied a lot in size, one possible explanation for the similar performance is that something like an 80/20 rule is in effect. That is, 20% of the definitions are doing 80% of the work and the remaining definitions do not contribute much to the final results. For the hand-crafted dictionary, we found that 18% of the definitions accounted for 80% of the instantiated concept nodes, and 28% of the definitions accounted for 90% of the instantiated concept nodes (when processing all 1700 MUC-4

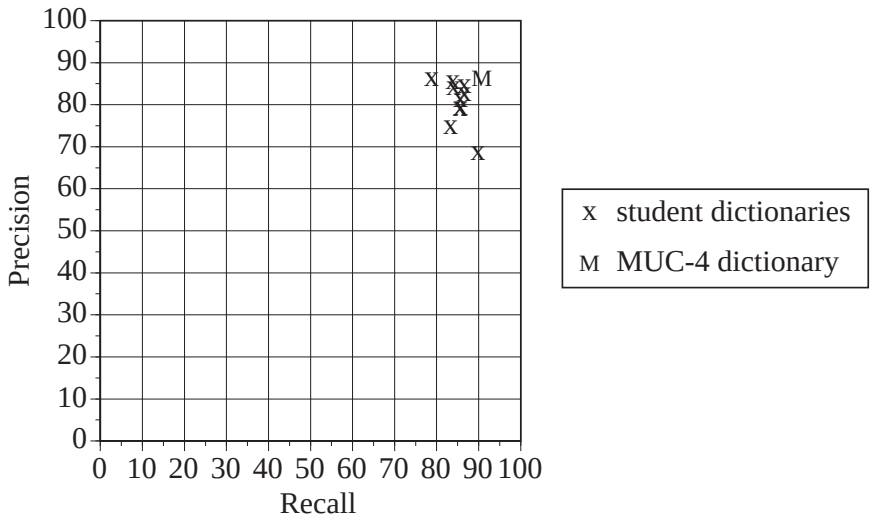

Fig. 30. Recall and precision scores for text classification

texts). These statistics are questionable because the number of times that a concept node fires does not necessarily indicate how much it contributed to the final scores, but they suggest that some definitions are more important than others, and that dictionaries produced by different people will probably contain similar subsets of the most important definitions.

4.2 An Experiment with Domain Experts in the Joint Ventures Domain.

The second experiment involved two government analysts who manually filtered a dictionary produced by AutoSlog for the joint ventures domain [33]. In contrast to the previous experiment, the government analysts had no background in natural language processing at all, or any experience with CIRCUS or the UMass/MUC-5 system. However, the analysts were considered to be experts in the joint ventures domain because they were among those who manually encoded the answer key templates for the MUC-5 corpus [28]. This experiment represents a more realistic example of how dictionaries would likely be constructed for new domains. It is more realistic to expect to find people who are experts in a particular subject, than to find people who are experienced in natural language processing (much less CIRCUS in particular). Furthermore, the analysts were motivated to generate good dictionaries. The analysts were evaluating a tool that they might use in the future, while the students were completing a homework assignment that was graded pass/fail. Before they began filtering, we gave the analysts a 1.5 hour tutorial explaining how AutoSlog works and how to use the interface.

AutoSlog proposed 3167 concept node definitions for the joint ventures domain, but the analysts were only available for two days and we did not expect them to be able to review all 3167 definitions in this limited time. So we cre-

ated an "abridged" version of the dictionary by eliminating entity and product/service definitions that were proposed infrequently by AutoSlog [13], and we removed the morphology module from the interface. The resulting "abridged" dictionary contained 1575 concept node definitions. Analyst A took approximately 12.0 hours to do the filtering and Analyst B took approximately 10.6 hours.

We compared the analysts' dictionaries with the MUC-5 dictionary that was filtered by an experienced UMass researcher. To ensure a fair comparison, we created an abridged version of the UMass dictionary by removing all of the definitions that were not among the 1575 given to the analysts, and removing all of the definitions spawned by the morphology module. The abridged MUC-5 dictionary was therefore based on exactly the same definitions given to the analysts, but was filtered by a UMass researcher. Figure 31 shows the number of definitions proposed by AutoSlog for each information type, and the number of definitions in each filtered dictionary.

CN Type	#proposed by AutoSlog	#kept (MUC-5 Abridged)	#kept (Analyst A)	#kept (Analyst B)
entity	688	311	357	423
facility	80	20	16	55
ownership percent	174	91	117	91
person	243	119	149	52
product/service	316	76	152	44
revenue rate	19	14	12	16
revenue total	30	22	15	26
total capitalization	25	14	13	22
TOTAL	1575	667	831	729

Fig. 31. Comparative Sizes of the Analysts' Dictionaries

To evaluate the dictionaries, we removed the original MUC-5 dictionary from the UMass/MUC-5 system, and plugged in the analysts' dictionaries and the abridged MUC-5 dictionary. [14] Finally, we scored each system on the Tips3 blind test set that was used for the MUC-5 evaluation. The Tips3 collection

[13] This was based on the frequency counts described in Section 2.4. We removed all entity definitions that were proposed < 2 times and all product/service definitions that were proposed < 3 times. We eliminated entity and product/service definitions simply because they dominated the dictionary.

[14] One complication was that the UMass/MUC-5 system includes two modules, TTG and Maytag, that used the original MUC-5 concept node dictionary for training (see [21]). Ideally, we should have retrained these components for each run with the new dictionary. We did retrain TTG, but we did not retrain Maytag. It is unlikely that this had a significant impact on the relative performance of the dictionaries, but we are not certain of its exact impact.

contained 282 texts. Figure 32 shows the scores for each system.

TIPS3	Recall	Precision	F-measure	ERR
Abridged MUC-5	18	51	27.06	83
Analyst A	19	47	27.39	83
Analyst B	20	47	27.89	83

Fig. 32. Scores for the analysts' dictionaries

All three dictionaries achieved similar scores. Overall, both of the analysts' dictionaries achieved slightly higher f-measures than the MUC-5 dictionary. The error rates (ERR) for all three dictionaries were identical (see [28] for a description of the error rate measure), but the dictionaries filtered by the analysts achieved slightly higher recall and lower precision than the MUC-5 dictionary. One possible explanation is that the UMass researcher was not as knowledgeable about the domain and was therefore conservative about accepting only the definitions that looked obviously reliable. The analysts were much more familiar with the domain and probably kept additional patterns that were familiar to them (but not necessarily as reliable).

Despite the fact that the composition of the dictionaries varied quite a bit, the final scores were remarkably similar. Even though they had no background in text processing, the analysts' produced dictionaries that performed at least as well as the one created by a UMass researcher. This is further evidence that we are probably seeing something like an 80/20 rule in effect, where a core subset of the definitions shared by most of the dictionaries do most of the work. This result has important implications for system development: if possible, data should be presented to users in order of expected impact. Many systems are built in a limited time frame, and users don't have time to review all of the potentially useful data. With respect to AutoSlog, we could rank the concept nodes based on frequency. The concept nodes that were proposed most frequently by AutoSlog would be presented to the user before concept nodes that were proposed only a few times.

5 Conclusions

We have shown that AutoSlog can produce effective dictionaries for information extraction in multiple domains. Most information extraction systems rely on a dictionary of extraction patterns that must be hand-coded for each domain [12,15,1]. However, a system called PALKA [14] has also been developed to automatically acquire patterns for information extraction. The output produced by PALKA is similar to the output produced by AutoSlog, but PALKA should be distinguished from AutoSlog along several dimensions.

First, PALKA is given a set of generic frames and keywords for the domain by a user. In contrast, AutoSlog discovers the trigger words for case frames on its own. Second, PALKA relies on the semantic features associated with words to identify the extraction patterns. AutoSlog does not use a semantic feature dictionary at all.

Other researchers have worked on the general problem of automated dictionary construction. FOUL-UP [10] was one of the earliest AI systems that automatically learned the meanings of unknown words. The POLITICS [3] system also contained a mechanism for learning definitions for unknown words. Both FOUL-UP and POLITICS learned information about unknown words by examining contextual expectations derived from other words in the sentence. RINA [13] is a language acquisition system that used multiple examples and a variety of knowledge sources to create dictionary entries for unknown words. All of these systems started with a "partial lexicon", and assumed that most of the words in the sentence were already defined. Definitions for new words were constructed based on the definitions of other words in the sentence or surrounding context. 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 or a statistical part-of-speech tagger (e.g., POST [36]).

One exception is recent work on automatically deriving knowledge from on-line dictionaries (see [7,25]). This research applies syntactic and lexical patterns to the entries in an on-line dictionary to derive semantic relationships between words. Although the goals are different, this work is similar in spirit to AutoSlog because syntactic rules are applied to text to extract semantic relationships. Their results lend independent support to the idea that semantic information can be acquired automatically without a lot of external knowledge.

Since AutoSlog creates dictionary entries from scratch, it can be viewed as a one-shot learning system. The closest points of comparison in the machine learning community are explanation-based learning (EBL) systems [6,24]. Explanation-based learning systems 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., [8,29,35]). Inductive learning systems typically require both positive and negative training instances to produce a target representation.

As input, AutoSlog requires an annotated training corpus for the domain and a few hours of manual filtering. However, NLP systems often rely on other types of tagged corpora, such as part-of-speech tagging or phrase structure bracketing (e.g., the Brown Corpus [9] and the Penn Treebank [22]). Furthermore,

corpus tagging for AutoSlog is less demanding than other forms of tagging because it is smaller in scope, and only the targeted information needs to be tagged (in contrast to syntactic tagging for which every word or phrase must be tagged). However, we are currently working on a new version of AutoSlog, called AutoSlog-TS, that does not need detailed text annotations at all but just a corpus of preclassified texts [34]. We have also shown that information extraction can be used to achieve high-precision text classification [32], so the dictionaries produced by AutoSlog are useful for other language processing tasks as well.

We have shown that novices can use AutoSlog effectively with only minimal training. When building systems for automated knowledge acquisition and rapid prototyping, it is important to remember that the ultimate users of these tools will be domain experts, not computer scientists. Tools that are accessible only to fellow researchers will be of limited use in the real world. Therefore we believe it is important not only to evaluate the performance of a system when tested by researchers, but also to evaluate the performance of a system when tested by potential users.

In summary, AutoSlog is a major contribution toward making information extraction systems portable across domains. AutoSlog was the first system to automate the process of dictionary construction for information extraction, and substantially reduces the knowledge-engineering bottleneck for building information extraction systems. AutoSlog demonstrates that some types of domain-specific semantic knowledge can be acquired automatically using only an appropriate training corpus. We believe that research in automated dictionary construction is crucial for natural language processing systems to become practical for real-world applications, and AutoSlog is a significant step in that direction.

Acknowledgments

We would like to thank Wendy Lehnert for her help in setting up both of the experiments described in Section 4, David Fisher and Jon Peterson for designing and programming the AutoSlog interfaces, and Stephen Soderland and Jon Peterson for being the humans in the loop. This research was supported by NSF Grant no. EEC-9209623, State/Industry/University Cooperative Research on Intelligent Information Retrieval and NSF Grant MIP-9023174.

References

[1] D. Ayuso, S. Boisen, H. Fox, H. Gish, R. Ingria, and R. Weischedel. BBN PLUM: Description of the PLUM System as Used for MUC-4. In *Proceedings*

of the Fourth Message Understanding Conference (MUC-4), pages 177–185, San Mateo, CA, 1992. Morgan Kaufmann.

[2] J. G. Carbonell. *Subjective Understanding: Computer Models of Belief Systems*. PhD thesis, Research Report 150, Computer Science Department, Yale University, 1979.

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

[4] R. E. Cullingford. *Script Application: Computer Understanding of Newspaper Stories*. PhD thesis, Research Report 116, Computer Science Department, Yale University, 1978.

[5] Gerald DeJong. An Overview of the FRUMP System. In W. Lehnert and M. Ringle, editors, *Strategies for Natural Language Processing*, pages 149–177. Lawrence Erlbaum Associates, 1982.

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

[7] William Dolan, Lucy Vanderwende, and Stephen D. Richardson. Automatically Deriving Structured Knowledge Bases from On-Line Dictionaries. In *Proceedings of the First Conference of the Pacific Association for Computational Linguistics*, pages 5–14, 1993.

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

[9] W. Francis and H. Kucera. *Frequency Analysis of English Usage*. Houghton Mifflin, Boston, MA, 1982.

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

[11] Philip J. Hayes and Steven P. Weinstein. 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*, pages 49–64. AAAI Press, 1991.

[12] Jerry R. Hobbs, Douglas Appelt, Mabry Tyson, John Bear, and David Israel. SRI International: Description of the FASTUS System Used for MUC-4. In *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, pages 268–275, San Mateo, CA, 1992. Morgan Kaufmann.

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

[14] J. Kim and D. Moldovan. Acquisition of Semantic Patterns for Information Extraction from Corpora. In *Proceedings of the Ninth IEEE Conference on Artificial Intelligence for Applications*, pages 171–176, Los Alamitos, CA, 1993. IEEE Computer Society Press.

[15] G. Krupka, P. Jacobs, L. Rau, L. Childs, and I. Sider. GE NLTOOLSET: Description of the System as Used for MUC-4. In *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, pages 177–185, San Mateo, CA, 1992. Morgan Kaufmann.

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

[17] W. Lehnert, C. Cardie, D. Fisher, J. McCarthy, E. Riloff, and S. Soderland. University of Massachusetts: Description of the CIRCUS System as Used for MUC-4. In *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, pages 282–288, San Mateo, CA, 1992. Morgan Kaufmann.

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

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

[20] W. Lehnert, C. Cardie, D. Fisher, E. Riloff, and R. Williams. University of Massachusetts: MUC-3 Test Results and Analysis. In *Proceedings of the Third Message Understanding Conference (MUC-3)*, pages 116–119, San Mateo, CA, 1991. Morgan Kaufmann.

[21] W. Lehnert, J. McCarthy, S. Soderland, E. Riloff, C. Cardie, J. Peterson, F. Feng, C. Dolan, and S. Goldman. UMass/Hughes: Description of the CIRCUS System as Used for MUC-5. In *Proceedings of the Fifth Message Understanding Conference (MUC-5)*, pages 277–291, San Francisco, CA, 1993. Morgan Kaufmann.

[22] M. Marcus, B. Santorini, and M. Marcinkiewicz. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.

[23] M. Mauldin. Retrieval Performance in FERRET: A Conceptual Information Retrieval System. In *Proceedings of the 14th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 347–355, 1991.

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

[25] S. Montemagni and L. Vanderwende. Structural Patterns vs. String Patterns for Extracting Semantic Information from Dictionaries. In *Proceedings of the Fourteenth International Conference on Computational Linguistics (COLING-92)*, pages 546–552, 1992.

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

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

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

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

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

[31] E. Riloff. *Information Extraction as a Basis for Portable Text Classification Systems*. PhD thesis, Department of Computer Science, University of Massachusetts Amherst, 1994.

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

[33] E. Riloff and W. G. Lehnert. A Dictionary Construction Experiment with Domain Experts. In *Proceedings of the TIPSTER Text Program (Phase I)*, pages 257–259, San Francisco, CA, 1993. Morgan Kaufmann.

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

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

[36] R. Weischedel, M. Meteer, R. Schwartz, L. Ramshaw, and J. Palmucci. Coping with Ambiguity and Unknown Words through Probabilistic Models. *Computational Linguistics*, 19(2):359–382, 1993.