



This breakthrough method for sorting through reams of text, linking relevant information while ignoring the irrelevant, has stimulated research into natural language processing and promises practical text-analysis applications.

Jim Cowie

Wendy Lehnert

Information Extraction

There may be more text data in electronic form than ever before, but much of it is ignored. No human can read, understand, and synthesize megabytes of text on an everyday basis. Missed information—and lost opportunities—has spurred researchers to explore various information management strategies to establish order in the text wilderness. The most common strategies are information retrieval (IR) and information filtering [4]. A relatively new development—information extraction (IE)—is the subject of this article.

We can view IR systems as combine harvesters that bring back useful material from vast fields of raw material. With large amounts of potentially useful

information in hand, an IE system can then transform the raw material, refining and reducing it to a germ of the original text (see Figure 1).

Suppose financial analysts are investigating production of semiconductor devices (see Figure 2). They might want to know several things:

- Which chemicals are deposited to produce insulating layers;
- The thickness of the layers;
- The temperature at which the layers are formed; and
- Who uses the process.

Such information is frequently available in newspaper and journal articles, and IR systems can collect the articles with relevant text. IE starts with a collection of such texts, then transforms them into information

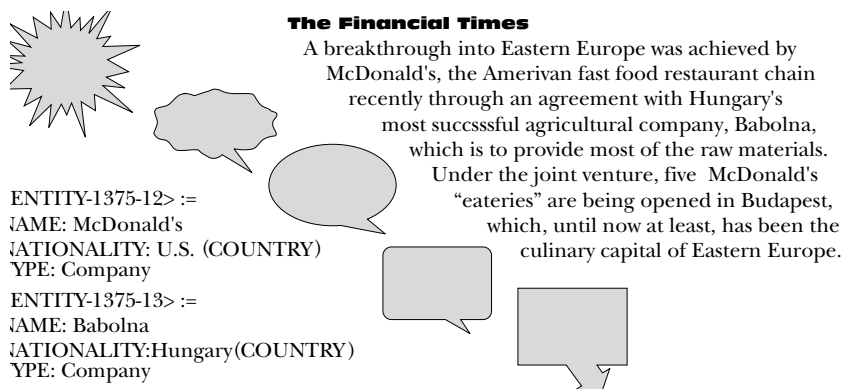


Figure 1.
IE transforming a text's structures

that is more readily digested and analyzed. It isolates relevant text fragments, extracts relevant information from the fragments, and then pieces together the targeted information in a coherent framework. For example, an article may discuss chemical gases, temperatures, processes, and material specifications, but only one or two of these items may be of interest to the human analyst. The goal of IE research is to build systems that find and link relevant information while ignoring extraneous and irrelevant information.

IE has numerous potential applications. For example, information available in unstructured text can be translated into traditional databases users can probe through standard queries. Suppose we want to track the profits of U.S. forestry companies and compare them with those of European forestry companies. Relevant information includes company name, company nationality, the fact that the company is in the forestry industry, and the amount and currency of its profits. An IE system that tracks news releases in this area, updating a database as the information becomes available, can help detect trends as soon as public announcements are made. Other IE systems might process news events, including meetings of important people, formation of new companies, and announcements of new products. Meanwhile, IR systems could benefit from the extracted information to construct sensitive indices more closely linked to the actual meaning of a particular text.

Accurate IE is the long-term goal of the research described in this article. Today, however, IE systems deal only with specific types of texts and are only partly accurate.

IE and Natural Language

From the viewpoint of natural language processing (NLP), IE is attractive for many reasons, including the following:

- Extraction tasks are well defined.
- IE uses real-world texts.
- IE poses difficult and interesting NLP problems.
- IE performance can be compared to human performance on the same task.

The fact that IE systems can be evaluated and compared to human benchmarks yields special opportunities for NLP researchers. It also means government financial support for semiannual Message

Understanding Conferences (MUCs) [7, 8] and for ARPA's Tipster Text Program, which coordinates multiple research groups and government agencies, seeking to improve IR and IE technologies [9]. Within five years, we can expect a number of practical applications.

Along with the positive interest in IE, hard questions are being asked by members of the NLP community, especially whether IE diverts scarce resources and talent to software development activities of no value to long-term NLP research. To address this issue, we conducted a survey of attendees at MUC-4 in 1992 (the most recent such survey) [8]. Selected excerpts from that survey are published here for the first time. One line of thought—shared by the authors of this article—maintains that IE systems are a key factor in encouraging NLP researchers to move from small-scale systems and artificial data to large-scale systems operating on human language.

A slightly different type of text extraction—knowledge extraction—is also being pursued at a few research sites. Knowledge extraction systems face many of the problems faced by IE systems. But knowledge extraction systems seek to deduce a rule base or a domain model on the basis of technical text. Such efforts include a strong machine-learning component, in addition to the NLP component [19]. The knowledge base they hope to extract is frequently designed to drive an expert system [13] or case-based reasoner. This is generally a more ambitious undertaking than the IE discussed here, in which the task is like filling out a form. Early IE system development is exemplified by the following systems:

- One of the first reported IE systems, which operated on texts of unrestricted topics, was implemented by Gerald deJong [11, 12]. Using a newswire network as its data source, deJong's program, called FRUMP, monitored a newswire using simple scripts designed to cover news stories. FRUMP sought to match each new story with a relevant script on the basis of keywords and conceptual sentence analysis. FRUMP was a semantically oriented system that used domain-specific expectations to instantiate

```

<doc>
<REFNO> 000019641 </REFNO>
<DOCNO> 3560177 </DOCNO>
<DD> November 25, 1991 </DD>
<SO> News Release </SO>
<TXT>

```

Applied Materials, Inc. today announced its newest source technology, called the Durasource, for the Endura™ 5500 PVD system. This enhanced source includes new magnet configurations, giving the industry's most advanced sputtered aluminum step coverage in sub-micron contacts, and a new one-piece target that more than doubles target life to approximately 8000 microns of deposition compared to conventional two-piece "bonded" targets. The Durasource enhancement is fully retrofittable to installed Endura 5500 PVD systems. The Durasource technology has been specially designed for 200mm wafer applications, although it is also available for 125mm and 1s0mm wafer sizes. For example, step coverage symmetry is maintained within 3% between the inner and outer walls of contacts across a 200mm wafer. Film thickness uniformity averages 3% (3 sigma) over the life of the target.

```

</TXT>
</doc>

```



```

<TEMPLATE-3560177-1> :=
  DOC NR: 3560177
  DOC DATE: 251192
  DOCUMENT SOURCE: "News Release"
  CONTENT: <MICROELECTRONICS_CAPABILITY-3560177-1>
  DATE TEMPLATE COMPLETED: 021292
  EXTRACTION TIME: 5
  COMMENT: "article focuses on nonreportable target source but reportable info available"
  /"TOOL_VERSION: LOCKE.3.4"
  /"FILLRULES_VERSION: EME.4.0"
<MICROELECTRONICS_CAPABILITY-3560177-1> :=
  PROCESS: <LAYERING-3560177-1>
  MANUFACTURER: <ENTITY-3560177-1>
<ENTITY-3560177-1> :=
  NAME: Applied Materials, INC
  TYPE: COMPANY
<LAYERING-3560177-1> :=
  TYPE: SPUTTERING
  FILM: ALUMINUM
  EQUIPMENT: <EQUIPMENT-3560177-1>
<EQUIPMENT-3560177-1> :=
  NAME_OR_MODEL: "Endura(TM) 5500"
  MANUFACTURER: <ENTITY-3560177-1>
  EQUIPMENT_TYPE: PVD_SYSTEM
  STATUS: IN_USE
  WAFER_SIZE: (200 MM)
              (121 MM)
  COMMENT: "actually three wafer sizes, third is error 1s0mm"

```

Figure 2.
Completed templates
for semiconductor
sources

event descriptions based on scriptal knowledge.

- An even earlier project—before 1970—for extracting useful information from text was directed by Naomi Sager of the Linguistic String Project group at New York University [23]. Sponsored by the American Medical Association, this work sought to convert patient discharge summaries (filled out in English) to a form suitable for use as input to a traditional Conference on Data Systems Languages (CODASYL) database management system.
- In 1980, DaSilva and Dwiggins extracted satellite-flight information from reports produced by monitors around the world [10], but the system was restricted to single sentences and lacked a methodology for extracting complete event descriptions.
- Zarri started working on IE systems in the early 1980s [25]. The texts he used described the activities of various French historical figures. The system sought to extract information about the relationships and meetings between these people.
- In 1981, Cowie developed an IE system that extracted canonical structures from field-guide descriptions of plants and animals [6]. The system used simple information to populate a fixed-record structure.

The main difference between the systems developed in the 1980s and those developed in the 1990s is the large amount of time and energy needed to col-

lect relevant documents and to create sets of templates (or keys) to produce test corpora consisting of texts and the associated “correct” answer. For example, although development of a text and its associated key by a human analyst for the Tipster microelectronics domain is expensive and complicated, these resources—text and associated keys—have made IE unique with respect to other natural-language task orientations (see Figure 2). Templates make it possible to evaluate the performance of IE systems while simultaneously playing an important role in developing the systems.

Common Ground

The first and second MUCs¹, called MUCK1 in 1987 and MUCK2 in 1989, were concerned with extracting information from a small number of short naval messages [24]. (The K in MUCK was dropped from the names of more recent conferences.) A change of topic and text type was made for the third and fourth conferences, called MUC-3 and MUC-4 [7, 8]. These conferences used newspaper and newswire texts on the general topic of terrorist incidents in several Latin American countries. A template structure and associated extraction rules were defined, and the participating groups generated filled-out templates for 1,000 texts. In addition, a semiautomatic scoring program was commissioned and another set of templates was independently produced for the evaluations. Participants had a set of training texts and templates; at each of the yearly MUC evaluations, they were presented with a new set of texts their systems were required to process. The participants scored their systems’ performance against the new templates. Any decisions involved in scoring were evaluated by an independent group [16].

MUC-3 and MUC-4 involved a largely stable group of about 17 research sites.² These groups met three times to discuss their progress, problems, and proposed plans in preparing for future system evaluations. There were dry-run meetings and evaluations of the MUC sites in January 1991, prior to MUC-3 in July 1991 and MUC-4 in June 1992. These meetings proved effective for sharing hard-won insights and lessons. Although traditional conferences and workshops are designed to facilitate similar communications, the common denominator of a shared application is its ability to act as a quick catalyst for cross-fertilizing ideas. Feeling pressure to demonstrate

progress, MUC sites are receptive to innovative ideas from competing labs, especially when the labs show superior performance. Cross-fertilization also helps reveal trends—partial parsing, shallow knowledge, and template mining—discussed later in this article.

DARPA’s interest in IE, inspired by the progress shown by MUC-3, prompted the funding of several large-scale IE projects using various approaches, although all worked on the same tasks. For example, the Tipster research initiative was intended to combine IR and IE, but in the first phase of development—1991-1993—research on each was carried out independently. For the past two years, the researchers from the IR and IE groups have met at monthly workshops, engaging in a valuable exchange of ideas, especially in tasks of common importance, including part-of-speech tagging and proper-name recognition.

Tipster encourages creation of systems that are portable and reconfigurable.³ Tipster designers chose two different domain areas: the business area of joint ventures and the technical area of developments in semiconductor fabrication techniques. Language independence and portability were emphasized by requiring creation of systems for each of two languages, English and Japanese. Four thousand texts and templates were required as training data, along with additional evaluation texts. The templates for Tipster were produced by analysts specially trained for the task. The resulting corpus is a unique NLP resource.

Before MUC and Tipster, creation of templates by human analysts for actual tasks were developed for only a limited number of texts. Now, the performance of Tipster’s human analysts [1] shows that if professional analysts were tested, they would probably earn grades of 60% to 80% for overall IE and accuracy—a lot lower than is intuitively expected. However, it may be less surprising in light of a task’s complexity. For example, the extraction rules for the Tipster joint-venture domain fills 38 pages, and some of the analyzed articles run up to 10 pages. Analysts must suspend their world knowledge and expertise and extract only the data that can be found in the text. The task itself is tedious, as any Tipster or MUC research group can confirm, and therefore ideally suited to automation.

Scoring and evaluating the performance of IE systems is still an area for research. Several measures were developed for MUC-3 and MUC-4 [16]. The two original measures used for MUC-3, based on the standard IR measures “recall” and “precision,” were computed

¹ Participants apparently took a lighthearted view of the first two Message Understanding Conferences. Their original acronym—MUCK—was changed to the relatively sober MUC in time for the third conference.

² MUC-4 participants: Bolt, Beranek and Newman, General Electric, General Electric/Carnegie Mellon University, Hughes Research Laboratories, Language Systems, McDonnell Douglas, MITRE, New Mexico State University/Brandeis University, New York University, Paramax Systems, PRC, SRA, SRI International, University of Maryland/ConQuest, University of Massachusetts, University of Michigan, University of Southern California.

³ The Murasaki project also encourages cross-language portability [3, 30] processing two languages—Spanish and Japanese. The domain consists of reported AIDs cases.

for each slot in the template, then used to produce various aggregate scores. Recall is the number of slot fills matched correctly divided by the total number of slot fills in the key. Precision is the total number of slot fills produced correctly divided by the total number of slot fills produced. MUC and Tipster scoring systems both have to resolve how the slots are aligned before scoring. The original MUC scoring program aligned the fields in the system's response and in the corresponding key to maximize the score. A weighted combination of recall and precision, called the F measure, has been used as the main measure since 1992. A new measure based on error rates at the MUC-5 conference in 1993 and on the final Tipster evaluations as the primary measure of performance.

In terms of recall and precision, the best performance of the MUC and Tipster systems is characterized by about 40% recall and 50% precision. In terms of speed, machine performance far exceeds human performance. The New Mexico State University Computing Research Laboratory's extraction system for Japanese microelectronics processes 100 texts in 30 minutes. The average analyst (as logged by the template-filling tool) takes 20 hours for the same task. The SRI system (reported at MUC-4) [8] processes 100 messages in 11 minutes.

A further sign of IE viability is the reality of commercial systems, including the following:

- The earliest IE system to be deployed as a commercial product, called ATRANS, was designed to handle money-transfer telexes in international banking [18]. It used sentence analysis techniques similar to those in FRUMP and exploited the fact that the content of money-transfer telexes is highly predictable. ATRANS demonstrated that relatively simple language processing techniques are adequate for IE applications narrow in scope and utility.
- JASPER is an IE system that extracts information from reports on corporate earnings [1], achieving robust NLP capabilities through template-driven techniques for language analysis operating on small sentence fragments. JASPER development depends on a manual inspection of the system in action, along with ongoing system evaluations using representative test sets.
- SCISOR is a prototype incorporating IE into an integrated system [21] that uses partial analysis of the text to carry out its processing. A filtering process selects only articles on corporate mergers and acquisitions, extracting information on target and suitor companies and dollar-per-share amounts. These items are stored in a knowledge base that handles queries.

IE is still a novelty outside the U.S., although several notable IE research projects are found in other countries. For example, Zarri's work continues in Paris. Udo Hahn at the University of Freiburg in Ger-

many is researching the extraction of information on microprocessor systems from magazine articles [14]. His main emphasis is the topic structure of text. A group at Fiat in Milan, Italy uses IE to process short reports by car mechanics [5], and a group at Sussex University in England developed a system that processes police traffic report messages [20]. In these last two systems, the input messages are short and there is no need for discourse analysis within a message, although both systems must resolve references between individual messages.

Further evidence of international IE interest is that NEC (in Japan), the University of Sussex, and the University of Manitoba (Canada) all participated in MUC-5 in 1993 in Baltimore.

IE Trends

Recent IE research emphasizes the importance of many neglected areas of NLP while demonstrating that simple methods are adequate for many tasks involved in analyzing text. Some of these methods are examined in the following sections. Figure 3 shows the basic components of a typical IE system. Most IE systems depend on off-line components to produce the data and rules that direct on-line processing. Automation and speed-up of these components are critical for rapidly porting systems to new domains. The speed of the system is another critical factor in iterating to achieve high levels of accuracy through a fast development cycle.

Two aspects of preprocessing seem to be common to many IE systems under development:

- Part-of-speech tagging programs (discussed later in the Trainable NLP Systems section) to allow preliminary recognition of phrasal units in sentences; and
- Special-purpose rules to recognize the semantic classes of phrasal units, including company names, places, people's names, currencies, and equipment names.

Real text is rich in proper names and expressions for dates, values, and measurements. These phrasal units are used productively, usually posing no problem to human readers. However, when looking at the Tipster rules for a company name, we find ambiguity where a company is reported in such forms as: "Sumitomo of Malaysia announced today . . ." For example, is "of Malaysia" part of the name or not?

The nature of these units requires that they not be held in a phrasal lexicon and that an automatic system recognize them, either through context or through patterns of subunits in the text. IE systems should recognize such constructions accurately. For some applications, a database of texts and lexicon lists taken from these texts provides adequate coverage. The same recognition capabilities can also support IR needs.

Tipster and all MUC efforts provide participants large amounts of proper-name information through

extraction from templates and standard resources (the Tipster Gazetteer includes 250,000 place names) and resources generated and shared by the participants, such as company-name lists.

This richness of lexical resources causes its own problems, as lexical ambiguity also applies to proper names. For English texts, upper- and lower-case information can help, but much newswire material is still transmitted in upper case only. Because most syntactic tagsets include proper nouns, they can be used to detect boundaries. Such delimiters as titles, (e.g., Mr. and Dr.) and company designators (e.g., Corp. and Inc.) can also be used to good effect. A large lexicon can allow unknown words, in conjunction with other indicators, to form part of a proper name pattern. Performance appears to range from 40% to 90%, depending on domain and techniques used.

Getting By with Minimal Syntax

Partial parsing refers to a style of sentence analysis that does not require a full syntactic parse of a sentence to pursue semantic analysis. Proponents of semantically oriented sentence analyzers have argued for 20 years that partial parsing is viable, but relatively little research has been conducted along these lines. Traditional pursuits within the linguistic community have favored full syntactic sentence analysis, a view that has dominated computational linguistics for the past three decades.

Unfortunately, a full syntactic analysis of every sentence in every text is too computationally demanding in the face of fast throughput demands. Sites attempting to generate full syntactic parse trees find themselves trapped inside a system development cycle too slow to support adequate experimentation and feedback loops. Syntactic sentence analyzers typically operate in polynomial time and tend to get bogged down with sentences containing more than 20 to 30 words. These cycle-bound systems have motivated researchers to reconsider their requirement for complete syntactic parse trees.

Some sites have preserved their syntactic analysis while working to separate relevant sentences from irrelevant sentences to reduce the amount of throughput being processed. Other sites could relax their syntactic constraints to produce fragments of parse trees or incomplete parse trees. Still other sites have abandoned their full syntactic analyzers altogether to pursue true partial parsers never intended to provide a full syntactic analysis in the first place. In particular, finite-state approximation grammars have been shown to provide a surprisingly effective engine for partial-parsing system design [8]. One way or another, some ability to skim a text is preferable to approaches that uniformly

apply ambitious sentence analysis to all portions of the source text.

Illustrating this trend, several MUC-4 participants reported the following:

- “Prior to MUC-3, no one would have considered that parsing fragments would work, now everyone does it.”
- “It was encouraging to see how well the SRI group did using a system with little more than the expressive power of regular languages.”

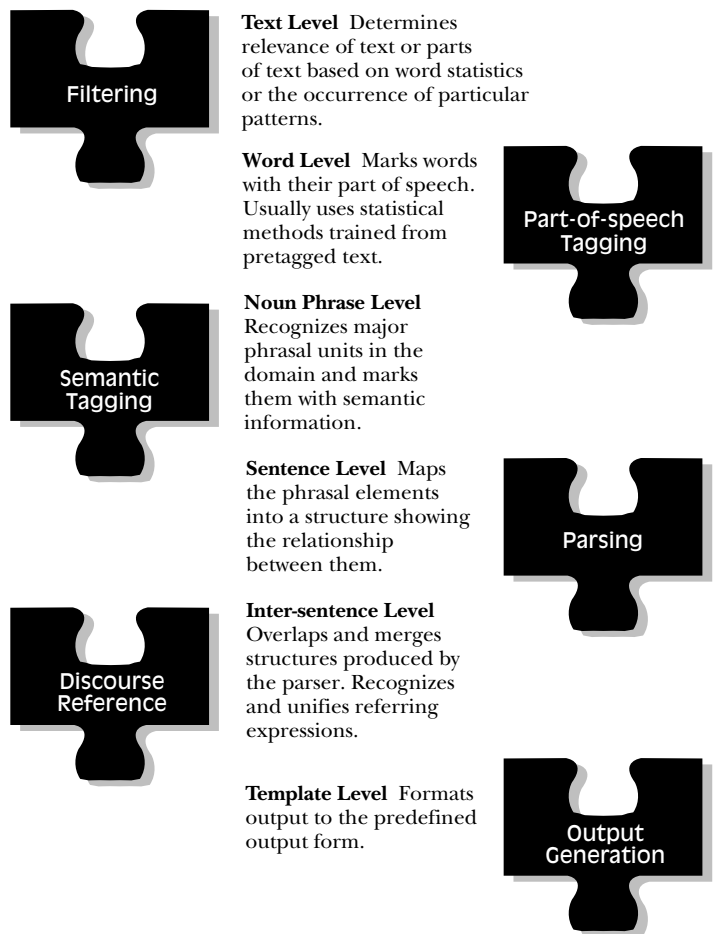


Figure 3. Typical IE system components

- “We learned to apply linguistic knowledge with successive relaxation. This was a direct result of skimming, in which we literally skipped over information in the process of text extraction. This methodology is being used for robust syntactic analysis.”

The trend toward partial-parsing techniques does not mean that partial parsing is adequate for all NLP applications. On the other hand, it is probably fair to say that a thoughtful re-examination of mainstream assumptions about syntactic sentence analysis is in order. One can no longer assume a priori that full syntactic sentence analysis is required as a component in all NLP architectures.

The Power of Shallow Knowledge

Shallow knowledge is a term coined by MUC-3 participants to describe their approaches to the knowledge engineering bottleneck. Less knowledge engineering means faster development cycles, less development expense, and more time for data collection, data analysis, and internal evaluations. However, some amount of knowledge engineering is still necessary. The key questions are what kind and how much? The previous section discussed ad hoc, domain-specific, finite-state machines—and how well they were doing. Terms like “ad hoc” and “domain-specific” tell us that knowledge engineering is present. When we talk about minimizing a system’s knowledge engineering, we mean minimizing the effort associated with knowledge acquisition.

At MUC-4, we heard reports from two relatively successful sites—SRI and the University of Massachusetts (UMass)—focusing on the knowledge engineering issue [8]. In both cases, the type of knowledge being acquired was domain specific and ad hoc. But the acquisition methods allowed large amounts of this knowledge to be collected with minimal effort. Both of these systems exploited large amounts of shallow knowledge.

Shallow knowledge is easily criticized because it is not reusable in other systems and is useful only in the specific application for which it was created. But there is a tacit assumption that deserves careful examination—called the significant-price assumption—that says that each knowledge unit must be obtained at some cost to the user and that this cost is significant enough to create knowledge-engineering bottlenecks. This is not unreasonable, given the experience of expert systems and other knowledge-intensive system engineering efforts. The significant-cost assumption is the primary rationale behind the CYC Project, which justifies the effort through the hope that the knowledge created will be useful to many systems [17].

How does shallow knowledge relate to the significant-cost assumption? Shallow knowledge may be so cheap to acquire that it is cost-effective to treat it as a disposable artifact. If shallow knowledge also proves to be adequate for the application at hand—even if only for a single domain, a single task, and a single system—shallow knowledge obviates the significant-cost assumption, and we need not worry about knowledge-engineering bottlenecks. If there is no knowledge-engineering bottleneck, the cost of knowledge acquisition becomes one more piece of

overhead associated with creating a large computer program, indistinguishable from the costs associated with code optimization, documentation, and effective user interfaces.

The UMass/MUC-4 system derived its shallow knowledge from the text corpus so its dictionary could handle sentence analysis with respect to interactions between syntax and semantics. The domain is Latin American terrorism. Each of the following UMass/MUC-4 shallow-knowledge examples is followed by a rule derived automatically for inclusion in the dictionary:

- A group of armed individuals wearing ski masks robbed a businessman. Rule: The direct object of “robbed” (active voice) is the victim of a robbery.
- The Assistant Secretary disappeared on Jan. 12. Rule: The subject of “disappeared” (active voice) is the victim of a kidnapping.
- The group was traveling in a four-wheel-drive vehicle. Rule: The object of “in” after traveling (active voice) is the target of an attack.
- Reported that dynamite sticks were hurled from a car. Rule: The subject of “hurled” (passive voice) is the instrument of an attack
- A bomb was placed in the parking lot of Government House. Rule: The subject of “placed” (passive voice) is the instrument of a bombing.

These examples show that there is little in the system of value to an NLP system operating in other domains. However, a dictionary based on these definitions—375 of them—helped the UMass/MUC-4 system achieve fair performance levels in the official MUC-4 test runs. These definitions work so well because they are shallow and thoroughly specific to the task at hand. Moreover, it is possible to compile such a dictionary with minimal manual knowledge engineering. Using a UMass-developed dictionary construction tool called AutoSlog, UMass researchers can create concept-node dictionaries for a new domain in less than 10 hours of human interaction. Experiments conducted by the UMass team show that undergraduates and first-year graduate students with little or no background in NLP can create AutoSlog dictionaries that perform almost as well as project researchers.

Note, too, that a complete IE system still contains other modules subject to knowledge engineering bottlenecks. Although the UMass concept-node dictionary is just one component requiring knowledge engineering, it demonstrates how shallow knowledge can be powerful enough to drive an effective IE system, while being cheap enough to violate the significant-cost assumption.

Discourse Analysis

Discourse analysis involves three problems:

- Noun-phrase analysis, which refers to the problem of recognizing appositives and other complex noun

- phrases and the semantic analysis of noun phrases;
- Co-reference resolution, which refers to the problem of knowing when a new noun phrase refers back to a previously encountered referent; and
- Relational links recognition, which is needed to structure memory tokens in an associative network containing links known to be important for supporting IE requirements.

(A discussion by several MUC-3 participants appears in the MUC-3 proceedings [7].) Of the three problems, co-reference resolution is by far the most challenging. Important entities are likely to be mentioned many times and may never be described twice by the same noun phrase. Co-reference resolution is responsible for knowing when different descriptions refer to the same thing. Co-reference resolution is sensitive to structural features of the overall text, as well as to semantic features, correct sentence analysis, and complex noun-phrase analysis. Any propagation of errors associated with identification of these features is readily apparent when we attempt to recognize co-referent noun phrases.

A major challenge in discourse analysis is the question of domain-dependent and domain-independent heuristics. It is one thing to implement a rule base that handles discourse analysis for a specific application. It is quite another to create domain-independent capabilities allowing us to port discourse-analysis heuristics from one domain to another. Although any given rule in a rule base can be categorized as domain-dependent or domain-independent, the interactions of many rules in a rule-based architecture are difficult to analyze. For example, if a rule is removed and replaced by a new rule, the system must know whether the new rule should be invoked in the same place or in a different place. Multiple rules interact with one another, creating pathways through the rule base. It may not be so easy to replace one set of domain-dependent rules with a new set if the interactions between the domain-dependent, and domain-independent rules have to be re-evaluated throughout the entire rule base.

Trainable NLP Systems

NLP developers researching IE functions are trying to build large-coverage systems readily portable to new domains by people who are not computational linguists. One promising avenue of inquiry is determining whether it is possible to automate acquisition of the data and rules needed for a new language or domain by training the system to perform certain parts of its task. This option is more viable with the increased availability of large corpora, human-tagged texts (both syntactic and semantic), and machine-readable dictionaries.

Three systems using statistical methods prove the value of training in IE—for part-of-speech tagging, for recognizing relevant texts, and for semantic pat-

tern acquisition. They are discussed in the following paragraphs.

Part-of-speech tagging using statistical models of text is a common IE system feature. For example, Bolt, Beranek and Newman developed its Part-Of-Speech Tagger (POST) system using texts from the *Wall Street Journal* that are hand-tagged by the NLP group at the University of Pennsylvania. With the addition of some fix-up rules to improve performance on unknown words, POST performance is impressive, about 95% correct. This result is comparable to results from other statistically based methods. Statistically based software for Japanese text, which involves the additional problem of recognizing word boundaries, was developed at Kyoto The program, called JUMAN, segments Japanese text while tagging the text with part-of-speech tags and semantic-feature tags.

Text tagging involves several variations, but all depend on the computation of the frequency of a word/part-of-speech (POS) pair being followed by any other word/POS pair; sometimes the frequency of tri-grams (three words in succession) is used. Once these frequencies are established, the estimated probability of each pair can be computed. For new sentences, the sequence of pairs with maximum probability is computed through a hidden Markov model. Information provided through this method removes syntactic ambiguities, allowing production of fragments, such as nominal compounds, and simplifying subsequent parsing.

M

any IE systems include a document filter as part of their pre-processing [1, 8, 15], allowing the system to ignore part or all of a text if it fails to satisfy certain conditions. Although texts processed by IE systems are normally provided by an IR system, these texts are not guaranteed to be relevant. In particular, keyword-based Boolean IR can find irrelevant documents that can cause problems for a robust NLP system. For example, seven of the 1,000 Tipster microelectronics articles discuss potato chips. Some texts need to be excluded for more subtle reasons. For example, an extraction rule for MUC-4 specified an attack by terrorists on a military target that was not considered a terrorist incident—the rationale being that terrorist incidents are directed at civilian targets. Many texts in the MUC corpus are actually reports of speeches in which terrorism is condemned, not reports of actual incidents. Finally, a long text may contain relevant information in only a few paragraphs; the ability to ignore the rest would be useful, although not necessarily essential if the IE system is specific enough in its own processing to ignore this material.

Several systems have experimented with proba-

TYPE Filled by the type of the entity, using **COMPANY** as a default. The **TYPE** of an aggregate **ENTITY** is **COMPANY** if the elements are all companies (e.g., “three Japanese auto companies” is **COMPANY**), “local interests” is **OTHER**; a family or tribe is **OTHER**, not **PERSON**, and **ASEAN** is **OTHER**, not **GOVERNMENT**.

COMPANY Any profit-making or nonprofit legal entity, including universities, partnerships, corporations, proprietorships, consortiums, enterprises, and government-owned corporations.

PERSON A person who appears to be operating on his or her own behalf if there is no overt reason to believe that the person is incorporated.

GOVERNMENT The government of a country, state, or municipality. The rule about not inferring information from the name of an entity does not apply to Government entities. “The Government of Indonesia” should result in a **TYPE** of **GOVERNMENT**. Text that says, “IBM announced a joint venture with China” should lead to the inference of **TYPE OF GOVERNMENT** unless a co-referential expression elsewhere in the document indicates a different **TYPE**.

OTHER An entity that does not fit these criteria (e.g., “a group of investors,” “the Apache Indian tribe of North Dakota,” **OPEC**, the Medellín cartel, and international bodies, such as **ASEAN** and **NATO**). This category is not equivalent to “I don't know.” Some entities do not meet the criterion of “legal entity,” as in the case of Medellín Cartel.

ENTITY_RELATIONSHIP Object

If CompanyA is the **CHILD** of CompanyB and CompanyC, the parents are implicitly defined as partners, so a separate **ENTITY_RELATIONSHIP** object need not be defined. If entities are clearly involved but the exact relationships are not clear, an **E_R** should not be used. If no explicit child (i.e., joint venture) company is formed, only the **PARTNERS** relationship should be used.

In cases of mutual ownership, if the reciprocal ownership is reported as equal, **PARTNER** is the type; if one owns a greater percentage of the other, a **SUBORDINATE** relationship is reported. The conditions for reportability of entities must still be met.

Instantiate this object if either there are two or more fills in the **ENTITY1** slot or there is at least one fill each in **ENTITY1** and **ENTITY2**.

Figure 4.
Extracts from Tipster's
joint-venture fill rules

bilistic models of word frequency to compute the probability that a text is relevant. These systems depend on the availability of relevant and irrelevant

documents. The templates can deliver this information, and keys containing information are from relevant texts. To detect relevant paragraphs, the strings in the keys can be matched back against the text.

Given relevant texts or paragraphs, it is still necessary to find the distinguishing words and to compute their relative frequency in relevant and irrelevant texts. The words need not appear frequently in either document type, but they must appear in one document type more than in the other. This comparison allows computation of the probability that a document is relevant based on the frequency of occurrence of a document's distinguishing words.

This process works if appropriate data is available. It also allows a more IE task-specific filter to be applied to the texts while requiring minimal human intervention for setting up the process.

Automated Knowledge Acquisition

Mining the templates refers to any effort that systematically exploits target template encoding during system

development. A number of MUC sites have attempted this since 1991. For example, a (partial) dictionary of locations in Latin America could be compiled on the basis of the location slots in the target templates. Other useful data can also be compiled from the same source, like all the noun phrases in the corpus that describe a bombing target and a list of terrorist organizations. Various statistics help establish correlations and probability distributions. If a system needs to decide whether a kidnapping took place in country A or in country B, and the user has a database showing a significantly larger number of kidnappings in A than in B, the user might consider resolving

the uncertainty according to statistical probabilities.

At MUC-3, only one site (Hughes Research Laboratories) used templates as a major component in its development effort [7]. Mining the templates was initially dismissed by many NLP researchers, yet only a year later, this reluctance seemed to evaporate.

Asked what was most surprising about MUC-4, one participant said, “The extent of usage of development templates for training.” A question about the role of engineering prompted another participant to say, “It appears that almost every system, ours included, made extensive use of the key templates to guide system development.” Asked whether the MUC effort revealed anything about text extraction that could not be learned some other way, another respondent said, “The large amount of input, in the form of texts, and prescribed output, in the form of key templates, were extremely helpful for both knowledge acquisition and testing.”

Such comments should motivate us to think about using development templates in new ways. It may be too expensive to employ a human analyst to generate templates, annotate text, or do whatever else developers dream up. But it is still useful to think about a system development framework in which nontechnical people, who lack experience in computer pro-

gramming, can generate materials instrumental in a larger system development cycle.

The Extraction Problem

Defining the templates containing information extracted from texts is a difficult and complex problem. For example, the templates for MUC-3 and MUC-4 employed flat record structures with

- Efficient run times;
- Efficient knowledge engineering;
- Efficient system development interfaces; and
- Efficient system development environments.

The idea that good software development always requires good software engineering is worth repeating in the IE context, because NLP researchers do

“The only path to success in text extraction leads through the corpus . . .”

cross-references between fields. Multiple templates could be generated for any text. The templates for Tipster are in the form of “objects” that can contain pointers to other objects. The MUC templates are easier for humans to read and fill in, but the Tipster templates reduce redundancy and group-related information. Figure 2 shows a typical Tipster template, which holds data in named slots—some in the form of strings from the text. Much of this information is converted to normalized forms according to detailed rules needed for both system construction and ensuring consistency among the human analysts performing the extraction. These rules can be long and complex. See Figure 4 for two short extracts from the Tipster joint-venture rules.

Defining the IE task for the rich kind of information expected for Tipster and MUC is a complex task. The definer must understand the subject domain, the kind of information regularly available in the texts being used, and the capabilities of NLP systems. In some ways, the definitions for Tipster and MUC were made to test and extend the capabilities of NLP systems. The Computing Research Laboratory developed tools for creating Tipster’s key templates. The first of these went through six months of version changes, many due to the modification of the template definition. The definition problem must be addressed more fully before such systems are readily implemented. One approach is to define much simpler template structures that extract a minimal, useful subset of the information in the texts.

IE Software Development

The need for a fast development cycle has been apparent since 1992. In 1991, many sites seemed satisfied to have a working system operating with some level of competence. In 1992, it became clear that more rapid progress would help allow systems to be revised within a systematic and efficient system development cycle. When system evaluations are crucial to the development effort, performance data must be collected quickly and easily. Fast development cycles depend on four key system features:

not always think of software engineering as a critical NLP research weapon. As NLP researchers become involved in corpus-driven research, speed and efficiency suddenly matter when a large text corpus has a central role in a research program. Asked about this, a MUC-4 participant said, “The only path to success in text extraction leads through the corpus. Successful approaches must be fast enough to run through a large number of stories because that is the only way to tell if a technique works in practice.”

This is not to say that IE from text is achievable through superior software engineering alone. The point is that whoever builds a large software system inevitably runs up against software engineering concerns. Failing to handle them well harms the overall effort accordingly.

The competitive/cooperative community of MUC sites shares a number of features, perhaps the most remarkable being the speed with which promising ideas emerge, propagate, and evolve as a form of community property. Asked what was most surprising about MUC-4, one participant said, “I was surprised to see how the success of one site could influence the work of another site in such a tight time frame. This rapid cross-fertilization would never happen in any other context.” Another said, “The ability of a competitive task to make people change paradigms.” And another said, “The level of cross-fertilization of systems. People try to do things that without MUC they probably wouldn’t try—or maybe even think of. This was obvious by the striking similarities between NLP systems of MUC participants.”

Something about the MUC infrastructure apparently accelerates research as a community process. However, proper attribution is not always given. But the tension that would normally arise from a failed attribution seems to be offset by the fact that so many people are watching so closely. Within the context of this community, ideas operate as community property. Apart from the logistical problems,

this accelerated research is paid for in part by the participating research sites. This pace is not just the result of common denominators and shared experience. MUC participants really put themselves on the line, and there is clearly pressure to perform in the public evaluations.

If the pace of the evaluations is fast enough and the pressure to improve great enough, researchers may be unwilling to take chances. A researcher could opt for a conservative research plan promising incremental improvements over a riskier endeavor. This potential danger is inherent in performance evaluations and must be handled with great sensitivity by evaluation sponsors, researchers, and observers alike. Although the risk is present, the benefits of shared tasks and open competition are undisputed.

The Future

Major questions about IE are unanswered. For example:

- Can the technology be made sufficiently accurate? The answer depends on the application, but less than 90% precision is unlikely to satisfy analysts.
- Can the cost of producing IE systems be reduced to acceptable levels? The actual cost of building an IE system is unknown, but estimates at a recent Tipster workshop suggest it should take from three to six person-months for a computational linguist to port a system to a new domain.
- Can IE systems be deployed by non-language specialists? Domain-specific approaches may be acceptable if systems can be readily tuned to a new domain by nonspecialists.

The impact of future IE technologies will be significant if quality systems can be produced at reasonable cost. The impact of IE on our understanding of natural language is already significant for many NLP researchers, and the influence of MUC evaluations on the NLP community cannot be underestimated.

Asksed whether their involvement in MUC taught them anything about text extraction they might not have learned otherwise, MUC-4 participants cited insights associated with discourse analysis, the effectiveness of partial-parsing techniques, the importance of fast development cycles, and awareness of NLP problems that cannot be ignored without significant consequence. Perhaps the most notable response was from a senior NLP researcher, who said, "The gap between natural language processing research and usable natural language technology is very, very large."

Note, too, that the NLP gap is not due only to the ignorance of NLP theorists or to inadequate theories

applied to difficult problems. The gap has more to do with the ways theoretical results address or fail to address specific problems as a technology evolves. IE researchers have discovered that some of their finest theoretical weapons are too sophisticated for the task at hand. For example, good results with respect to syntactic sentence analysis are available through relatively simple techniques. Meanwhile, other IE-related problems—like knowledge engineering and trainable systems—are not addressed by the NLP literature and deserve more attention.

The NLP community should review its research priorities while focusing on the IE challenge. Its long-standing preoccupation with syntactic sentence analysis leaves it ill-prepared to tackle the difficult problems of semantic-feature tagging, co-reference resolution, and discourse analysis. The knowledge engineering aspects of NLP deserve special status in a comprehensive research agenda for practical NLP systems. System development issues should be as important as the dictionary definitions.

Meanwhile, some problems associated with IE applications are easier to solve than we expected. The NLP community should separate the commonly addressed problems from the neglected problems and direct research accordingly. If the IE challenge forces researchers to review their research priorities in the context of practical NLP applications, IE's most important long-term result might be a stronger and more effective connection between NLP researchers and NLP system developers. \square

References

1. Andersen, P. M., Hayes, P. J., Heutner, A. K., Schmandt, L. M., and Nirenberg, I. B. Automatic extraction. In *Proceedings of the Conference of the Association for Artificial Intelligence* (Philadelphia, Penn.). 1986, pp. 1089–1093.
2. Aone, C., Blejer, H., Flank, S., McKee, D., Shinn, S. The Murasaki Project: Multilingual natural language understanding. In *Proceedings of the DARPA Spoken and Written Language Workshop*. 1993.
3. Ayuso, D., Bobrow, R., McLaughlin, D., McLeer, M., Ramshaw, L., Schwartz, R., and Weishedal, R. Towards understanding text with a very large vocabulary. In *Proceedings of the DARPA Spoken and Written Language Workshop*. (Hidden Valley, Penn.) Morgan Kaufmann, 1990, pp. 354–358.
4. *Commun. ACM* 35. Special Section on Information Filtering, Terry, D. and Loeb, S., Eds., (Dec. 1992), 26–81.
5. Ciravegna, F., Campia, P., and Colognese, A. Knowledge extraction from texts by SINTESE. In *Proceedings of the 14th International Conference on Computational Linguistics (COLING92)* (Nantes, France) 1992, pp. 1244–1248.
6. Cowie, J.R. Automatic analysis of descriptive texts. In *ACL Proceedings, Conference on Applied Natural Language Processing* (Santa Monica, Calif.), 1983, pp. 117–123.
7. DARPA. *Proceedings of the 3d Message Understanding Conference (MUC-3)* (San Diego, Calif.), Morgan Kaufmann, 1991.
8. DARPA. *Proceedings of the 4th Message Understanding Conference (MUC-4)* McLean, Va., Morgan Kaufmann, 1992.
9. DARPA. *Proceedings of the Tipster Text Program (Phase I)* Fredricksburg, Va., Morgan Kaufmann, 1993.
10. DaSilva, G. and Dwiggin, D. Towards a Prolog text grammar. *SIGART* 72 (1980).
11. DeJong, G. F. Prediction and substantiation: A new approach to natural language processing. *Cognitive Sci.*, 3 (1979), 251–273.

12. DeJong, G. F. An overview of the FRUMP system. In *Strategies for Natural Language Processing*. W.G. Lehnert and M.H. Ringle, eds. Erlbaum, Hillsdale, N.J., 1982, pp. 149–176.
13. Delannoy, J.F., Feng, C., Matwin, S., and Szpakowicz, S. Knowledge extraction from text: Machine learning for text-to-rule translation. In *Proceedings of the Workshop on Machine Learning Techniques and Text Analysis, ECML93* (Vienna). 1993.
14. Hahn, U. On text coherence parsing. In *Proceedings of the 14th International Conference on Computational Linguistics (COLING92)* (Nantes, France). 1992, pp. 25–31.
15. Jacobs, P.S. and Rau, L.F. SCISOR: Extracting information from on-line news. *Commun. ACM* 33, 11 (1990), 88–97.
16. Lehnert, W. and Sundheim, B. A performance evaluation of text analysis technologies. *AI Mag.* 12, 3 (1991), 81–94.
17. Lenat, D.B. and Guha, R.V. *Building Large Knowledge-Based Systems: Representations and Inference in the CYC Project*. Addison-Wesley, Reading, Mass., 1989.
18. Lytinen, S. and Gershman, A. ATRANS: Automatic processing of money transfer messages. In *Proceedings of the 5th National Conference of the American Association for Artificial Intelligence*. IEEE Computer Society Press, 1993, pp. 93–99.
19. Matwin, S. and Szpakowicz, S. Text analysis: How can machine learning help? In *Proceedings of the 1st Conference of the Pacific Association for Computational Linguistics (PACLING)* (Vancouver, Canada). 1993, pp. 33–42.
20. Mellish, C., Allport, A., Evans, R., Cahill, L.J., Gaizauskas, R., and Walker, J. The TIC message analyzer. *Tech. Rep. CSR P 225*, Univ. of Sussex, 1992.
21. Rau, L. Extracting company names from text. In *Proceedings of the 7th Conference on Artificial Intelligence Applications* (Miami Beach, Fla.). 1991.
22. Riloff, E. and Lehnert, W. Classifying texts using relevancy signatures. In *Proceedings of the 10th National Conference of the American Association for Artificial Intelligence* (San Jose, Calif.). 1992, pp. 329–334.
23. Sager, N. *Natural Language Information Processing: A Computer Grammar of English and its Applications??*. Addison-Wesley, Reading, Mass., 1981.
24. Sundheim, B.M. and Chinchor, N.A. Survey of the message understanding conferences. In *Proceedings of the DARPA Spoken and Written Language Workshop*. 1993.
25. Zarri, G.P. Automatic representation of the semantic relationships corresponding to a French surface expression. In *ACL Proceedings, Conference on Applied Natural Language Processing* (Santa Monica, Calif.). ACL, 1983, pp. 143–147.

About the Authors:

JIM COWIE is a research specialist in and deputy director of the Computing Research Laboratory at New Mexico State University. His research interests include natural language processing, particularly for information extraction. **Author's Present Address:** Computing Research Laboratory, Box 3CRL, New Mexico State University, Las Cruces, NM 88003; email: jcowie@nmsu.edu

WENDY LEHNERT is a co-principal investigator in the Center for Intelligent Information Retrieval in the Department of Computer Science at the University of Massachusetts. Her research interests include natural language processing and cognitive models of human thought processes. **Author's Present Address:** University of Massachusetts, Department of Computer Science, Amherst, MA 01003; email: lehnert@cs.umass.edu

Permission to make digital/hard copy of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage, the copyright notice, the title of the publication and its date appear, and notice is given that copying is by permission of ACM, Inc. To copy otherwise, to republish, to post on servers, or to redistribute to lists requires prior specific permission and/or a fee.
