*AI Magazine,* pp. 81-94. Fall 1991.

# A Performance Evaluation of Text Analysis Technologies

by

Wendy Lehnert
Department of Computer and Information Science
University of Massachusetts
Amherst, MA 01003

Beth Sundheim
Naval Ocean Systems Center, Code 444
271 Catalina Boulevard
San Diego CA 92152

`

Evaluation has become an important and pressing concern for researchers in AI. We need to reassure ourselves and our program managers that progress is taking place, and that our technology is indeed advancing according to reasonable metrics and assessments. The difficulties with evaluating AI systems are substantial and, to some extent, idiosyncratic, depending on the area of specialization. In an effort to evaluate state-of-the-art natural language processing systems, the Naval Ocean Systems Center (NOSC) has conducted three evaluations of English text analyzers during the last five years. This report describes the most recent and most sophisticated of these evaluations, the Third Message Understanding Conference (MUC-3)[1] This evaluation was sponsored by the Defense Research Projects Agency (DARPA), which plays a key role in sponsoring evaluations for other types of language interpretation systems, including performance evaluations for speech recognition carried out by the National Institute of Standards and Technology (Pallett 1990).

## Background and History

In June 1990, a call for participation went out to research laboratories in industry and academia. The intent was to bring together established natural language processing systems for the sake of seeing how they would each handle a uniform text comprehension task. The call emphasized the importance of having a "mature" natural language processing system ready to go. The short time frame associated with MUC-3 was not amenable to extensive system construction or exploratory experimentation on a major scale. A total of 15 sites completed the final evaluation: 12 industry sites and 3 university sites. The participating sites were Advanced Decision Systems, Bolt Beranek and Newman, General Electric, General Telephone and Electronics, Hughes Aircraft, Intelligent Text Processing, Language Systems, McDonnell Douglas, New York University, Planning Research, Stanford Research Institute, Synchronetics, the University of Massachusetts, a joint effort between the University of Nebraska and the University of Southwestern Louisiana, and UNISYS.

The final evaluation measured each system's ability to extract information about terrorist incidents from a test suite of 100 previously unseen news articles. A uniform representation for terrorist events was adopted by all the participating systems to facilitate scoring based on

[1]This report does not represent an official analysis of MUC-3, its methodology or results. The views expressed herein are solely those of the authors as participants in the evaluation and conference. The official conference report is available as (*** 1991) [ed: this should be a citation to the MUC-3 conference proceedings - we couldn't figure out how to cite this using the author guidelines ...].

pre-defined answer keys. Each system was evaluated in terms of how much correct information was extracted (recall), how much of the information extracted was correct information (precision), and how much superfluous information was extracted (overgeneration). All test materials were actual texts released by a government agency and none of the participating sites had access to the test materials prior to the test administration.

All MUC-3 systems were expected to process essentially undoctored news articles about Latin American terrorism. Proper processing meant understanding which texts were relevant under our working definition of the domain, and generating instantiated template representations for those articles deemed relevant to the domain. In this way, the MUC-3 performance evaluation simulated a highly realistic information extraction task corresponding to a real-life scenario in which terrorism specialists require automated assistance to keep up with a heavy workload of material coming in over the newswires.

All systems were required to operate in a fully automated fashion without human intervention. No restrictions were placed on the processing techniques that could be used as long as the techniques were fully automated. Indeed, one of the goals of MUC-3 was to bring together a highly diverse collection of approaches in an effort to make some meaningful comparisons across text processing technologies.

The ultimate focus of MUC-3 was the final performance evaluation of all participating systems operating on novel texts. MUC-3 produced an essentially high-level view of overall performance. It would be more significant from a scientific perspective if the performance task could also yield useful insights into particular aspects of text analysis capabilities, such as the ability to handle new words, traditionally problematic syntactic constructions, and ill-formed sentences, to name a few. As an experimental sidelight to the evaluation, an effort was made to evaluate system performance on selected aspects of the task to see whether the performance evaluation methodology could support a finer-grained analysis. Although some success was achieved in doing this for MUC-3, further refinement of the methodology is required.

The time table for MUC-3 extended from October 1990 to May 1991. This allowed each site six months for system development devoted to the MUC-3 domain and task orientation, in addition to time needed for corpus development, official testing, and other responsibilities. It became increasingly apparent that each additional month or week might yield a dramatic difference in the final evaluation. Although each site probably would have benefited from more time, it was important to have an end in sight to keep everybody on course. The schedule created some pressure to optimize efforts and identify areas of improvement that could be realistically pursued in a short time frame. For some sites, this resource-limited optimization problem was possibly as challenging as the natural language processing problems!

Those who survived the MUC-3 experience learned a lot about the strengths and weaknesses of their systems, and the kinds of work that go into a building a fully functional natural language processing system. Realistic task orientations in natural language processing entail a great deal of work that is not specific to language per se. Researchers whose work had been more narrowly focused probably operated at a disadvantage relative to researchers who had had a lot of experience building comprehensive systems. Because of this substantial engineering factor, it was difficult to bridge the gap between the final MUC-3 system evaluations and any conclusions one might want to draw concerning specific theoretical ideas in computational linguistics. At the same time, MUC-3 has been very successful at raising some fundamental questions about the nature of natural language processing as a challenge in complex system design. With the deepest roots of the natural language community in linguistics, one might reasonably wonder if the engineering aspects of natural language processing have received adequate attention and consideration.

## The Task: Understanding News Articles

The MUC-3 effort addressed information extraction from continuous text. More specifically, the MUC-3 systems attempted to analyze articles distributed by the Foreign Broadcast Information Service of the U.S. Government. The texts came from multiple news sources, such as newspaper articles, summary reports, and rebel communiques, as well as transcripts from speeches and interviews. To aid system development, 1300 of these texts were included in a development corpus. The linguistic phenomena present in these texts provided realistic challenges in terms of connected text, complex syntactic structures, and an open-ended vocabulary (especially with respect to proper nouns). See Figure 1 for a sample text from the MUC-3 domain.

**Figure 1 is not available at this time.**

Figure 1. A Sample of MUC-3 Input and Output. The text analyzer instantiates a copy of the generic output template for each terrorist incident described in the source text. Each system must determine the number of output templates to generate for each text. If the number generated by the system does not match the number of templates present in the answer key, there will be a significant loss of recall or some amount of overgeneration. Even at the level of individual templates, each system must determine how many and which slots to fill. Answer keys may be sparse or full depending on the information present in the source text.

The goal was to extract information about terrorist incidents from text and represent those incidents using a generic template for terrorist activities. Each system generated instantiations of the generic terrorism template by filling empty template slots with appropriate values derived from the input text. The generic template contained 18 possible slots, not all of which applied to any given incident type. See Figure 1 for an output template instantiation generated in response to an input text.

There were 24 possible incident types including eight basic types (kidnappings, murders, bombings, attacks, arson, etc.) plus two variations on each (for threatened incidents and attempted incidents). Approximately 50% of the texts that were made available were deemed irrelevant to the domain according to an extensive set of guidelines designed to define relevant acts of terrorism. For example, an attempt was made to exclude terrorist-like incidents conducted in the context of guerrilla warfare. Systems were expected to determine when a given text contained relevant or irrelevant information.

Template slots are filled with either a closed class of acceptable slot fillers (e.g., the 24 incident types), strings from the source text (e.g., the name of a human target), or a cross-indexed combination of both (e.g., the perpetrator confidence slot must make a judgement concerning the reliability of the perpetrator's identity). Some slots are predefined as inapplicable to certain incident types (e.g. a kidnapping won't have an instrument type), and a slot should be filled with a “null” symbol if no information about that slot is present in the source text.

Many texts require multiple template instantiations to represent multiple events, and multiple event descriptions are often interspersed throughout a typical source text. This places a significant emphasis on discourse processing in addition to sentence analysis. Sometimes information needed to fill a slot is present only via inference. For example, weapons are frequently mentioned with no explicit description of their use. Operating in context, it is often obvious that the weapons are instrumental to the acts being described, but any such connections must be inferred.

Although limited in representational generality, the target templates designed for MUC-3 are fully satisfactory as a means for evaluating systems designed to extract limited amounts of

information. Note that the representational complexity of this task is made tractable by the emphasis on relevant information extraction. Any irrelevant information present in a text can effectively be ignored. An in-depth text analyzer designed to handle all the information present in these texts would require a level of representational machinery far beyond the scope of MUC-3. The fact that the terrorism domain could be reasonably characterized in terms of a single generic template was crucial for the success of MUC-3.

## The Development Corpus: Text and Templates

The MUC-3 evaluation would not have been possible without the distribution of a large, 1300-text development corpus containing both news articles and their target template encodings (answer keys). Each participating site was required to contribute to the creation of this corpus during the early stages of the project, and the answer keys for the development corpus were released six months before the final evaluation. All participating sites contributed to the development corpus by generating template representations for some specified segment of the 1300 texts. Pairwise combinations of sites were expected to compare overlapping portions of their results and work out any differences that emerged, of which there were many. To our knowledge, the distribution of labor behind the MUC-3 development corpus is unprecedented in AI (at least outside Japan) and was a critical component in making this corpus possible.

In terms of its content, the development corpus offers a snapshot of Latin American terrorism during the period from May 1988 to May 1990. The most frequent acts of violence are murder (404 incidents) and bombings (270 incidents). Kidnappings are very common (92 incidents), and arson occurs with substantial frequency (44 incidents). The sentences are realistically complex, and the task of understanding connected text as well as individual sentences is thoroughly challenging. The full development corpus contains approximately 400,000 words comprised of 18,240 unique lexical items. It also contains approximately 15,600 sentences with an average sentence length of 27 words. Each text in the corpus contains an average of 12 sentences (roughly half a page).

It is easy to underestimate the amount of time needed to generate high-quality template representations for unconstrained texts. Sites operating with only one or two researchers were particularly stressed by this requirement. In retrospect, we estimate that it takes an experienced researcher at least three days to cover 100 texts and produce good quality template representations for those texts. This is an optimistic estimate which assumes familiarity with a stable set of encoding guidelines. Furthermore, we are not taking into consideration the psychological difficulty of doing this particular task for long periods of time - most people find it necessary to walk away from the problem periodically to take a break. Our estimate also finesses the fact that two people will seldom agree on the complete representation for a specific text. It is much better to have two or three people independently generating representations which can then be compared, discussed, and adjusted as needed. If we allow time for study, rest, discussion, cross-checking, and revision, the total time needed to prepare answer keys for 100 texts is realistically between two weeks and a month. Since many participating sites were working under somewhat less than ideal circumstances, the original answer keys for the MUC-3 development corpus were uneven in quality, and further refinement over the course of the evaluation was not sufficiently thorough to bring the corpus up to a consistently high level of quality.

As difficult and time-consuming as the creation of the development corpus was, participation in its creation served a very useful function for each site involved. It encouraged a cooperative, involved, participatory spirit, and it ensured that everyone mastered the template specifications before launching into system development. Without this first-hand experience in the task of text encoding, it would have been much more difficult for everyone to acquire a

reasonable level of encoding expertise and thereby maximize consistency in interpretation and implementation across all the sites.

Despite all our good efforts, the task of generating correct encodings was not always clear cut. Many questions about encoding conventions proved difficult to answer. Some of these problems were related to issues of inference:

> Q: If a bomb explodes in a house, but no one is home, should we specify the resident (if identified) as a target of the bombing?
>
> A: No. just specify the house as the target.
>
> Q: When students pick up stones in response to some action by authorities, should we assume they are threatening to attack the authorities?
>
> A: Yes.

Other problems were more straightforward but still needed some guidelines:

> Q: If we have a sabotage incident with multiple targets, should that be viewed as a single event or multiple events with individual targets?
>
> A: A single event.

Many questions pointed out a need for a precise definition of the MUC-3 task:

> Q: How old does an event have to be before it assumes "background" status and should therefore not be reported?
>
> A: Two months.

Generic events lacking specific details were supposed to be irrelevant to the domain. But knowing when we had enough detail was not so simple.

> Q: When someone threatens to kill ten unnamed judges, is the intended target specific enough to be reported?
>
> A: Yes.

Even when guidelines were relatively straightforward, such as a distinction between military targets (irrelevant) and civilian targets (relevant), events could create combinations of targets that were no longer easy to categorize:

> Q: Military targets are not relevant to the MUC-3 domain, but do civilians in a military building lose their civilian status and therefore become irrelevant as bona fide targets?
>
> A: No, in fact the military building also becomes a relevant target under those circumstances.

Scores of questions like these were raised, usually motivated by specific examples, and general heuristics were not always forthcoming. For example, the notion of a generic event defied absolute definition despite repeated and concerted efforts. When heuristics were

delineated, they were frequently arbitrary in nature, and valuable only to the extent that everyone could know about them and use them. Simply keeping track of all the shifting encoding conventions became a major undertaking. Without e-mail communications and strong leadership from the program committee, this ongoing dialog involving 15 isolated sites would never have been possible, much less productive. Many questions were ultimately left unanswered, but the corpus was created, and despite its imperfections, the participating sites considered it to be invaluable.

System Development

The complete development corpus was released to all participating sites in November, along with a semi-automated scoring program. The scoring program was used to facilitate system development and internal evaluations as well as scheduled group evaluations. The combination of the development corpus with the associated scoring program proved to be a tremendous resource. One could choose to concentrate on specific slot fillers and monitor a narrow band of progress, or one could attempt a broad assault on all the slots and get immediate feedback on which of those slots were doing well. The scoring program was the key to the relative success of MUC-3 versus previous evaluation efforts: participants could understand the metrics by using the scoring program, and scoring was vastly improved with respect to quality and consistency.

Four scoring metrics were devised to evaluate system performance:

RECALL reflects the completeness and the true positive rate of slot fillers present in the output templates (as a percentage of the total slot fillers in the answer keys). This metric shows the amount of correct and relevant data returned by a system relative to the total amount of correct and relevant data present in the input text.

PRECISION reflects the accuracy of the slot fillers present in the output templates (as a percentage of the total slot fillers in the output templates). A system that generates as much incorrect information as correct information is operating at a lower level of precision than a system that generates no incorrect information (100% precision).

OVERGENERATION reflects the amount of irrelevant information being generated by a system. Templates describing irrelevant incidents or slot fillers that shouldn't have been generated all contribute to the amount of overgeneration by that system.

FALLOUT reflects a false positive rate. This metric shows how much tendency a system exhibits toward assigning incorrect slot fillers as the number of potentially incorrect slot fillers increases. This metric is a trend measure.

In general, good system performance is manifest by high rates of recall and precision along with low rates of overgeneration and fallout.

The two metrics that provided the greatest feedback with respect to system performance were recall and precision. Overgeneration received some attention, independent of its penalizing impact on precision. But fallout was largely ignored, for a variety of reasons, including the fact that it could be computed only for certain slots, it was hard to understand, and the systems under evaluation did not seem to exhibit the behavior that fallout is intended to capture.

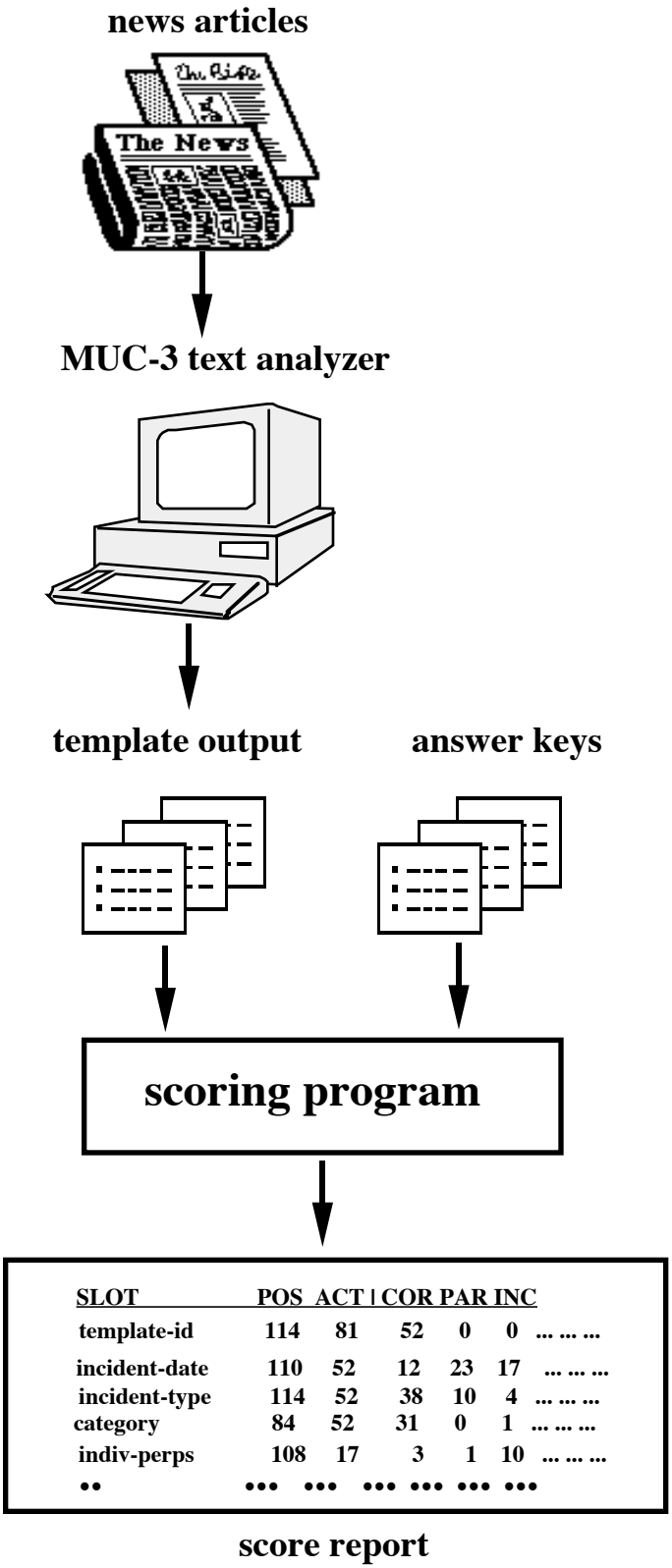


**Figure 2. The Experimental Design for MUC-3. This is the basic process design used for the MUC-3 evaluation. The same design was also used by individual sites for internal testing. Scores are generated by comparing system output with predefined answer keys. Slot fillers can receive no credit, full credit, or partial (50%) credit based on a set of stringent scoring guidelines. The scoring program can be run in a fully-automated mode for quick and easy evaluations, or with human interaction to enable careful scrutiny during an evaluation.**

The final evaluation called for systems to score as high as possible on both recall and precision. Sites were encouraged to consider processing modes which might effect some scoring trade-offs, especially trade-offs between recall and precision. One of our MUC-3 goals was to learn more about evaluation as we went, and the scoring program was eventually revised to reflect different ways of computing the basic metrics. It was recognized that application-specific requirements in real-life settings would dictate the relative importance of one measure over another. Consequently, no judgements were made as to whether the "best" systems were those that scored the highest on recall, highest on precision, or lowest on overgeneration.

For many of the participating sites, significant system deficiencies probably overwhelmed secondary concerns about scoring trade-offs. Systems that produced reasonable output at the sentence level had to be augmented with discourse analysis that could massage sentence-level output into target template representations. Systems that produced too much output had to invoke filters to separate the relevant information from the irrelevant information. Systems that were restricted in terms of syntactic complexity had to be scaled up to the complexity of the news articles. Systems that required extensive customization needed time to engineer those adjustments. MUC-3 provided ample opportunities to learn about system limitations, scale-up problems, and ways to track system development.

One aspect of system development that became somewhat troublesome was the question of application-specific processing. There were many opportunities to rely on default values and other heuristics that relate to application-specific guesswork. For example, if the sentence analyzer never identified a perpetrator, but a single known terrorist organization was mentioned at some point later in the text, one might reasonably guess that the named organization was in fact the correct perpetrator. If the confidence level for a perpetrator wasn't known, one might default to the value that appears in the development corpus with the highest frequency count. Reports at the preliminary test meeting suggested that recall rates as high as 15% might be attainable using application-specific defaults alone.

MUC-3 participants were not given any guidelines on how much application-specific processing should or should not be incorporated into their systems. For the sake of consistency, it was tempting to argue that no such processing be used. However, the prospect of legislating what is or is not application-specific is a non-trivial undertaking. In addition, it is basically impossible to draw a line between principled heuristics and cheap tricks, although we all think we know which is which when we see them. For example, the heuristic described above for identifying a perpetrator might constitute a cheap trick for an in-depth text analyzer that attempts to comprehend every sentence and extract all its representations one sentence at a time. But for a system that is filling slots on the basis of stochastic text profiles, the appearance of a lone terrorist organization in the text might constitute very strong grounds for slot filling behavior. One system's kludge might be another system's principled method. The best we could do was ask that everyone remain faithful to their own principles on this matter, whatever those principles might be.

Although it is difficult to say very much about system development that would apply to all of the participating sites, we can probably characterize the broad nature of the experience. At the end of phase 1, many systems were still quite immature, some of them in terms of system architecture and more of them in terms of their grasp of the task domain. Recall levels were extremely low in many cases (more than half the systems had an overall recall level less than 15%). These systems were just starting to climb the development curve they had to scale before a performance evaluation could be expected to shed any light on the strengths and weaknesses of their processing techniques.

Toward the end of phase 2, a number of the systems had matured to the point of handling the MUC-3 training texts more or less as planned. Because the amount of operational domain knowledge was still incomplete for many of these systems, some sites felt that their system's performance would continue to increase at a steady pace. In fact, all the sites reported that their development effort was effectively bounded by time: given another month, everyone predicted they would be able to increase their recall or precision by at least 5%. In some cases, new found levels of system maturity may have introduced a new type of risk that was not present during earlier stages of development.

For those sites that were no longer preoccupied with obvious problems, it was often difficult to say which component was holding back the scores the most, or which component might be the easiest to improve upon. If a system was having substantial difficulty with one particular template slot, was it advisable to focus a lot of attention on that one slot - or was it better to work for improvements in other areas that were causing less trouble? The problem of assessing cost/benefit ratios for various systems shortcomings gradually became more and more important as systems edged into states of greater maturity. Good or bad decisions about what to work on next could make all the difference, and there wasn't much time to recover from serious mistakes in this regard. More importantly, these strategic decisions had nothing to do with natural language processing per se - these were strictly system development issues. Here, more than ever, the engineering aspects of building a large complicated system intruded, and could easily overwhelm more theoretical considerations.

## System Evaluation: Phase 1

A preliminary evaluation of all participating systems was conducted in February, half way through the schedule, in an effort to identify problems that might be fixed or circumvented for the final evaluation. At the very least, this dry run gave everyone a chance to practice a serious test execution to make sure procedures were worked out and the systems were doing something. A meeting was held afterwards to give the participants an opportunity to see how everyone else was doing, and to talk about various problems and concerns.

The preliminary evaluation was based on a set of messages refered to as TST1. The TST1 test suite contained 100 previously unseen texts incorporating 5133 distinct tokens (where tokens correspond to words, numbers, and other identifiers in the text). There were 145 target templates associated with TST1. About 2% of the vocabulary present in TST1 was not present in the development corpus. The output templates produced by each system were analyzed using the scoring program, and overall performance was computed with respect to recall, precision, overgeneration, and fallout.

Summary score reports were based on the four metrics defined in the last section. The scores were computed across all 100 texts in TST1 for each slot individually and for the fillers of all the slots taken together. Averaging over all the sites, the TST1 score reports showed an average recall score of 15%, an average precision score of 56%, and an average overgeneration score of 21%.

At the time of the preliminary evaluation, many sites had not yet completed critical system components and many had not attempted to work on more than a few target template slots. Almost all the participating sites were aware of major problems that they hoped to get under control before the final evaluation. At the very least, this suggested that our so-called "mature" text processing systems were probably closer to an adolescent stage of development when MUC-3 first got underway. It is interesting to note that the TST1 score reports were not at all predictive of the final evaluation score reports. Performance evaluations are not meaningful for systems that have not reached some critical stage of developmental maturity.

In discussing the preliminary scoring runs, it became apparent that the scores we were generating were not the only way to compute performance. The main problem seemed to concern the treatment of spurious data, in particular the generation of spurious templates and its impact on the overall scores. The impact of template overgeneration was a modest penalty on overall precision, since the penalty was assessed by increasing the overgeneration score for just one slot, the template ID slot. This had the effect of providing a compromise view of overall performance, one that showed neither how well a system filled templates for just the relevant incidents, nor how much overgeneration a system incurred while reaching for higher recall. To provide this variety of views of overall performance, two new ways of computing the metrics were devised for use in phase 2.

Other adjustments were made to the encoding guidelines. Most notably, we decided to require cross-referencing in all the slots that referred to fills from other slots. This change increased the inherent difficulty of filling a template correctly, and it also necessitated further modifications to the original scoring program. In addition, the encoding changes rendered the development corpus obsolete and out of phase with the new scoring requirements. Consequently, one site immediately set to work to improve the consistency of the encodings and update them according to the new encoding guidelines.

With the preliminary evaluation behind us, each site was positioned to push as hard as possible for the next three months. Everyone predicted an improvement in their system's performance, but few sites were able to confidently predict exactly how much improvement to expect.

## System Evaluation: Phase 2

Alterations to the scoring program were completed after the preliminary evaluation and made available to the sites in a series of software updates. The final version of the scoring program was released about a month before the final evaluation. This final release incorporated three new scoring profiles to summarize overall system performance:

(1) MATCHED ONLY takes into account only those output templates that match the target template keys. Missing and spurious templates are not penalized except for the template ID slot. This provides a relatively generous measure of recall and precision based on only those incidents whose relevance was correctly determined by the system.

(2) ALL TEMPLATES takes into account all of the missing and spurious slots as well as the template ID slot. This is a relatively harsh measure of recall and precision that is sensitive to a system's inability to make correct relevance judgments.

(3) MATCHED/MISSING adds the totals as they appear in the score report columns. This profile assigns a penalty for each missing slot filler, but does not penalize spurious slot fillers except for the template ID slot. This was the original summary score used for phase 1, and it falls between MATCHED ONLY and ALL TEMPLATES in terms of relative harshness.

(4) SET FILLS ONLY reflects the totals as shown in the columns for only those slots that require a slot filler from a predefined and finite set of possible fillers. As such, this represents a score based on a subset of the slots evaluated the same way as under MATCHED/MISSING.

The final evaluation was based on the TST2 message set containing 100 previously unseen texts incorporating 4864 distinct tokens. There were 163 target templates associated with TST2.

Approximately 1.6% of the vocabulary present in TST2 was not present in the development corpus.

**Figure 3 is not available at this time.**

**Figure 3.** An Official TST2 Score Report (from the University of Massachusetts). The data in this report collapses slot fillers over the 100 TST2 texts and their associated answer keys. The most important columns are the four to the right which show the four basic metrics for evaluation. The four rows on the bottom collapse the slot filling data even further to compute overall evaluations across all the slot fillers as well as all the texts. The official MUC-3 scores appear in the MATCHED/MISSING row under the columns for recall, precision, and overgeneration.

MATCHED/MISSING was designated as the official scoring profile for MUC-3 because it represented the midpoint in scoring harshness and it was the only scoring profile used for phase 1. It is not the case that this one profile consolidates all the others or is necessarily the best profile for all purposes: a careful examination of all available scores should be made for any comprehensive comparison of the MUC-3 systems. A scatter plot of precision vs. recall over all the MUC-3 test sites (using MATCHED/MISSING) is presented in Figure 4.

**Figure 4 is not available at this time.**

Figure 4. The Official MUC-3 Scores for 15 Participating Research Sites. The partially automated scoring program allows humans to assign full or partial credit in cases where the scoring guidelines could not be reliably implemented by computer. To insure consistency across all sites, two individuals working together scored the TST2 output for all 15 sites. Even though the scoring program did most of the tedious scoring work, it took about one hour of computer-human interaction to generate the official TST2 score report for a single site.

The administration of TST2 was handled individually at each site according to strict testing guidelines. The test could be executed only once: systems that crashed were allowed to restart but were not allowed to reprocess any message that caused a fatal error. This stringent procedure resulted in scores for some sites that did not reflect true system capabilities. Given the complexity of these systems, careful consideration must be given to the role of system reliability in AI performance evaluations.

Despite the stringent testing, seven sites achieved at least 20% recall and 50% precision. Two systems exhibited recall scores over 40% with precision over 60%. Averaging over all the sites, the TST2 score reports showed an average recall score of 25% (vs. 15% for TST1), an average precision score of 54% (vs. 56% for TST1), and an average overgeneration score of 25% (vs. 21% for TST1). These comparative averages should be viewed as an extreme oversimplification of the data and should not be interpreted as a measure of the state of the art. They are also confounded by the fact that only 12 sites participated in TST1, one of those sites dropped out after TST1, and four new sites came on board after TST1 in order to participate in TST2. So in comparing the overall TST1 and TST2 scores, we are not really getting a true picture of how much ground was collectively covered by 15 sites in three months.

A factor frequently cited as a major troublespot was discourse analysis, when information derived from sentences is reorganized into target template instantiations. Five of the participating sites identified discourse analysis as their most compelling problem area and no site claimed to have a satisfactory discourse component. Discourse-level analysis has been a difficult area to pursue in a rigorous fashion because researchers have been limited to making

observations based on a small number of text examples. With a large corpus of the type available for MUC-3, processes at the discourse-level can now be studied more systematically.

The MUC-3 development corpus proved to be a crucial resource for the participating sites. Seven sites reported using at least 50% of the corpus to support various development efforts, and eight sites ran internal tests using the corpus at least once a week. In addition, the various ways that the corpus was used revealed a remarkable diversity of technologies represented by the participating systems. Some sites extracted information from the templates to build a domain-dependent lexicon, others used the texts to identify useful linguistic regularities, and one site used a portion of the development corpus as a case base for a case-based reasoning discourse component.

For another perspective on system diversity, one need only look at fundamental system features. Dictionaries ranged in size from 6,000 words to 60,000 words. Five systems generated syntactic parse trees for all the sentences in a text, and six systems never generated syntactic parse trees for any sentences. Nine systems used a formal sentence grammar of some sort, and six systems used no sentence grammars whatsoever. Remarkably, the four top-scoring systems spanned the spectrum on all three of these dimensions. One high-ranking system worked with a 6,000 word dictionary, no formal grammar, and no syntactic parse trees, while a close competitor operated with a 60,000 word dictionary, a syntactic grammar, and syntactic parse trees for every sentence encountered. Three additional systems used stochastic or inductive methods exclusively, and thereby served as comparative baselines for the natural language processing systems. When systems were ranked according to the highest combined recall and precision scores, the top eight systems were all natural language processing systems.

One measure of a task's difficulty is how well it can be performed by a well-understood technique originally designed for a simpler but related task. Motivated by such concerns, a volunteer from the MUC-3 program committee used an information retrieval technique, statistical text categorization, to generate templates and fill closed class slots for the TST2 messages. This produced a baseline of performance based on conditional probabilities derived from the development corpus with no additional linguistic knowledge or domain knowledge. Most of the MUC-3 systems outperformed this baseline. The categorizer was competitive, however, on the incident-type slot and the incident-category slot. This supports the intuition that these slots are most closely related to overall document content, while the other closed class slots demand significant attention to the internal structure of the texts.

Another baseline that we would like to establish is the performance level of human text encoders. Although we were not able to conduct a serious experiment along these lines, we have reason to believe that the recall rates of an experienced encoder when measured against a predefined answer key will not exceed 85%, and may actually be much lower. There is considerable disagreement between human encoders on a task of this complexity, and we have been observing the behavior of highly motivated researchers who are working very hard to adhere to the encoding guidelines. The performance of human encoders working on practical applications is likely to be characterized by even higher degrees of variance.

It is important not to equate the MUC-3 recall scores with standard grading curves where 90%-100% is very good, and 70%-80% is acceptable but not great. As far as information extraction is concerned, we may very well discover that highly trained people can manage no more than 80% recall when measured against a fixed set of answer keys. Without a carefully conducted study using human subjects and the MUC-3 test materials, we cannot say what recall rates constitute a human level of competence or how far our current state-of-the-art really is from human performance levels.

A complete compilation of TST2 data is available in the MUC-3 Conference Proceedings along with each participating site's own analysis of their test results and a system overview (*** 1991). In addition, all the output of each MUC-3 system on TST2 has been archived by NOSC and could serve as the basis for additional analyses. For example, it would be very interesting to isolate specific texts that all the systems handled well, none of the systems handled well, or only some of the systems handled well. By examining individual texts, we might attain a better understanding of text complexity than is possible from score reports alone. To see what is really going on, it is necessary to get underneath the overall numbers and see specific examples where a given approach is working or not.

## Conclusions

The very fact that MUC-3 took place represents a significant achievement of importance to the larger AI community as well as reseachers in natural language processing. First and foremost, we must emphasize the extreme complexity of the MUC-3 task orientation. Detailed information extraction from unconstrained text is as hard as any AI problem you can name. Domain coverage at the level of sentence analysis requires state-of-the-art knowledge engineering and the sentence analysis itself involves constraint satisfaction from multiple knowledge sources (typically syntax and semantics). Sophisticated reasoning at the level of discourse analysis requires more knowledge engineering along with powerful facilities related to abductive reasoning, temporal reasoning, and information fusion. The linguistic complexity of these texts is totally unconstrained: virtually every problem ever addressed by computational linguists can be found in this task. In addition to all that, strategies for fast scale up were critical given the strict time frame of MUC-3. No one can say we tackled a toy domain or a convenient application.

Given the substantial demands of MUC-3, the performance of the participating sites represents an impressive level of capability that may not have been attainable as recently as three years ago. A survey of the top-scoring MUC-3 systems reveals much about the state-of-the-art in text analysis:

- Text analysis techniques have progressed far beyond database interface applications and have demonstrated clear viability for information extraction from unconstrained text.

- Text analysis techniques incorporating natural language processing are superior to traditional information retrieval techniques based on statistical classification when applications require structured representations of information present in texts.

- Available techniques for semantic and syntactic sentence analysis are operating well and appear to be meeting the challenge of information extraction from unconstrained texts. More pressing difficulties are apparent at the level of discourse analysis.

- Experience with an operational data extraction system has shown that the throughput rates of human encoders can be increased by at least a factor of five. Although current levels of recall and precision may not yet be adequate for autonomous applications, MUC-3 supports the pursuit of applications for computer-assisted data encoding from unconstrained text.

• We still don't know the performance limitations of the MUC-3 systems. On average, less than 1 person/year of effort went into each of these 15 systems. It is impossible to say what another year of effort would yield.

• The top-scoring MUC-3 systems incorporate a diverse range of natural language processing techniques. With so many different approaches demonstrating viability, long term prospects for information extraction based on natural language processing are very promising.

MUC-3 provided the natural language processing community with a unique opportunity to put ideas on the line and see what could be done. It was gratifying to work on a project shared by other research sites, and to see everyone operating out of a sense of community as well as competition. Although our goals were ostensibly directed toward an evaluation of competing technologies, other benefits became apparent when we all met to discuss our work and progress. Multiple-site evaluations of this type strengthen intellectual contacts across otherwise disparate research groups, and work to facilitate a healthy cross-fertilization of ideas. As one participant observed at the preliminary evaluation meeting, "I have never been to another natural language meeting where I felt so intensely interested in what the other systems were doing and how well they were succeeding."

Participants in the MUC-3 performance evaluation agreed that sharing a common task had a profound effect on interactions among researchers. The MUC-3 conference presentations and proceeding papers were characterized by a high degree of openness and willingness to identify points of failure. The two MUC-3 meetings were very interesting for everyone in attendance, and the atmosphere was highly conducive to constructive communication. All in all, the experience of MUC-3 was intensely gratifying. The focus on comparable system performance was a welcome change from the usual impasses of theoretical claims and intellectual premises.

A MUC-4 evaluation is already being planned for the spring of 1992[2]. We hope to gauge progress after one more year of effort and to bring in new sites with innovative approaches to offer. MUC-4 will be based on the same domain and task orientation used for MUC-3, since these requirements proved to be sufficiently challenging for all the MUC-3 participants. There is more to learn more about available language processing techniques, and more to learn about evaluation. Participation will be open to research sites that have a viable text analysis system, and some domain-dependent lexical data will be provided to help new sites get started.

## Acknowledgements

A number of people were critical to the success of MUC-3 and deserve special acknowledgements. First and foremost, we must thank the members of the program committee for their ongoing involvement with a seemingly endless stream of responsibilities: Laura Balcom, Ralph Grishman, Jerry Hobbs. David Lewis, Lisa Rau, Beth Sundheim, and Carl Weir. In addition, John Sterling and Cheryl Kariya mediated over and solved many of our encoding problems during the creation of the development corpus. Pete Halverson wrote the scoring program which proved to be an essential component of MUC-3. George Krupka singlehandedly revised the entire development corpus in an effort to minimize inconsistencies across the corpus. Nancy Chinchor designed the linguistic phenomena tests and coordinated the effort to refine the evaluation metrics. Lynette Hirschman contributed information contained in this

[2]For more information about MUC-4 or access to the MUC-3 development corpus, contact Beth Sundheim at NOSC or via internet at SUNDHEIM@NOSC.MIL.

article, and provided us with many useful suggestions throughout. A draft of this article was discussed at the MUC-3 final evaluation meeting at which time many site participants made important corrections and suggestions. Any remaining inaccuracies are solely the responsibility of the authors. Finally, a special thank you is due to the DARPA sponsor of MUC-3, Charles Wayne, whose intellectual and emotional support was as critical as his financial support to the success of MUC-3.

## References

Pallett, D.S., Fisher, W.M., Fiscus, J.G., and Garofolo, J.S. 1990. DARPA ATIS Test Results. In Proceedings of the Speech and Natural Language Workshop, 114-121.. Hidden Valley, PA: Speech and Natural Language Workshop.

Sundheim, B.M. 1991. Proceedings of the Third Message Understanding Conference (MUC-3). San Diego, CA: Third Message Understanding Conference.

Sundheim, B.M. (in press). Third Message Understanding Evaluation and Conference (MUC-3): Phase 1 Status Report. In Proceedings of the Speech and Natural Language Workshop. Pacific Grove, CA: Speech and Natural Language Workshop.