TIPSTER/MUC-5
INFORMATION EXTRACTION SYSTEM EVALUATION

Beth M. Sundheim

Naval Command, Control, and Ocean Surveillance Center
RDT&E Division (NRaD)
Information Access Technology Project Team, Code 44208
San Diego, CA 92152-7420
sundheim@nosc.mil

INTRODUCTION

Three information extraction system evaluations using Tipster data were conducted in the context of Phase 1 of the Tipster Text program. Interim evaluations were conducted in September, 1992, and February, 1993; the final evaluation was conducted in July, 1993. The final evaluation included not only the Tipster-supported information extraction contractors but thirteen other participants as well. This evaluation was the topic of the Fifth Message Understanding Conference (MUC-5) in August, 1993. With particular respect to the research and development tasks of the Tipster contractors, the goal of these evaluations has been to assess success in terms of the development of systems to work in both English and Japanese (BBN, GE/CMU, and NMSU/Brandeis) and/or in both the joint ventures and microelectronics domains (BBN, GE/CMU, NMSU/Brandeis, and UMass/Hughes).

The methodology associated with these evaluations has been under development since 1987, when the series of Message Understanding Conferences began. The evaluations have pushed technology to handle the recurring language problems found in sizeable samples of naturally-occuring text. Designing the evaluations around an information extraction application of text processing technology has made it possible to discuss NLP techniques at a practical level and to gain insight into the capabilities of complex systems.

However, any such evaluation testbed application will undoubtedly differ in important respects from a real-life application. Thus, there is only an indirect connection between the evaluation results for a system and the suitability of applying the system to performance of a task in an operational setting. A fairly large number of metrics have been defined that respond to the variety of subtasks inherent in information extraction and the varying perspectives of evaluation consumers.

The evaluations measure coverage, accuracy, and classes of error on each language-domain pair, independently of all other language-domain pairs that the system may be tested on. With its dual language and domain requirements and challenging task definition, Tipster Phase 1 pushed especially hard on issues such as portability tools, language- and domain- independent architectures and algorithms, and system efficiency. These aspects of software were not directly evaluated, although information concerning some or all of them may be found in the papers prepared by the evaluation participants.

THE EVALUATION PROCESS

The Tipster contractors were allowed access to the training corpus (articles and hand-coded templates for a given language-domain pair) and associated materials (documentation, software resources, lexical resources) as they were being prepared over the course of Phase 1. The articles and corresponding hand-coded templates from the test corpus were held in reserve for use as blind-test materials during evaluation periods; new test sets were used for each evaluation. A description of the training and test corpora is contained in [1]. Those MUC-5 evaluation participants who were not Tipster contractors were allowed access to training materials in March, 1993, when major updates resulting from decisions made at the Tipster interim evaluation in February had been completed and permission for MUC-5 participants to use most of the copyrighted articles had been obtained. Table 1 identifies the MUC-5 evaluation participants and the language-domain pairs on which their systems were evaluated.

MUC-5 PARTICIPANT	CLASS OF PARTICIPATION	SYSTEM	EJV	EME	JJV	JME
BBN	Tipster	PLUM	X	X	X	X
GE/CMU	Tipster	SHOGUN	X	X	X	X
"	"	TEXTRACT			X	X
Language Sys., Inc.	non-Tipster	DBG		X		
MITRE	non-Tipster	ALEMBIC	X			
NEC (Japan)	non-Tipster	VENIEX				X
NMSU/Brandeis	Tipster	DIDEROT	X	X	X	X
NYU	non-Tipster	PROTEUS	X			
PRC, Inc.	non-Tipster	PAKTUS	X			
SRA Corp.	non-Tipster	SOLOMON	X		X	
SRI International	non-Tipster	FASTUS	X		X	
TRW	non-Tipster	DEFT	X			
Unisys-Paramax	non-Tipster	CBAS	X			
UManitoba (Canada)	non-Tipster	NUBA		X		
UMass/Hughes	Tipster	CIRCUS	X	X		
UMichigan	non-Tipster	LINK		X		
USouthern California	non-Tipster	SNAP	X			
USussex (UK)	non-Tipster	SUSSEX	X			

Table 1. MUC-5 Participation

The evaluation participants (Tipster and non-Tipster) were also provided with evaluation software, prepared via NRaD contract to SAIC, to help them monitor the performance benefits of alternative software solutions they were exploring in their research [9]. The evaluation software, corpora, documentation, and miscellaneous other resources were distributed primarily through electronic mail and electronic file transfer. Virtually every item was updated numerous times, and updates continued on some of them right up to the start of final testing. Personnel at the Consortium for Lexical Research (New Mexico State University) and the Institute for Defense Analyses played critical roles in making these materials available for electronic transfer.

At the start of the test week for each evaluation, the participants were supplied electronically with encoded test sets of articles, which they were to decode only when they were ready to begin testing. Testing was conducted by the participants at their own sites in accordance with a strict test protocol. After their systems processed the texts and produced the extracted information in the expected template format, the participants electronically transfered the templates to the Government for scoring.

Using the evaluation software prepared by SAIC, evaluators may score templates fully automatically (batch mode) or partially interactively (interactive mode). Since interactive scoring produces a slightly more accurate performance evaluation, scoring for the formal evaluations is usually done in that way. Some of the same analysts who hand-coded the answer-key templates prepared written guidelines for conducting the interactive scoring and did the scoring. SAIC conducted statistical significance tests on the 24-month/MUC-5 final test scores for the overall metrics of performance [3].

Table 2 summarizes the Phase 1 evaluations in terms of the test sets, participating sites, and primary evaluation metrics.[1] Since the JV template was especially complex, JV testing was done in two ways for each evaluation: (1) the core portion of the template, including the identification of tie-ups, entities, and relationships of entities within tie-ups, (2) the full template. The first microelectronics test was conducted at the 18-month point; up until a few weeks prior to that test, the Tipster contractors had had only a small portion of the EME and JME corpora available to them. The first JME evaluation (at 18 months) was conducted using all but the **<packaging>** objects.

The same test sets used for the Tipster 18-month evaluation were used for the MUC-5 dry run, with one exception: certain articles in the EJV test set had to be omitted because permission for non-Tipster MUC-5 participants to use those copyrighted articles had not been obtained. (Permission to use all but two of these

[1]This tabulation ignores the fact that the period of time covered by Tipster Phase 1 also included MUC-4. The Tipster Phase 1 contractors were evaluated for MUC-4 in the terrorism domain even as they were beginning their Tipster research and development in the Tipster domains [MUC-4].

sources were obtained in time for the MUC-5 final test.) The Tipster contractors did not participate in the dry run.

The primary performance metrics changed in the course of Phase 1. These are discussed below.

CHARACTERISTIC	12-MO TIPSTER TEST (SEP92)	18-MO TIPSTER TEST (FEB93)	MUC-5 DRY-RUN TEST (MAY93)	24-MO/MUC-5 FINAL TEST (JUL93)
Test sets	EJV core[2], full JJV core[1], full	EJV core[1], full JJV core[1], full EME full JME partial[3]	EJV[4] core[1], full JJV core[1], full EME full JME partial[2]	EJV core[1], full JJV core[1], full EME full JME full
Sites	Tipster[5]	Tipster[6]	Non-Tipster[7]	Tipster[5] Non-Tipster[6]
Primary metrics	Recall-precision	Recall-precision	Error	Error

Table 2. Tipster Phase 1 extraction system evaluations

Evaluation Criteria and Metrics

In assessing the performance of the information extraction systems, we are interested in knowing the classes of errors made and the circumstances in which those errors were made. We are also interested in performance from an applications perspective in terms of the completeness and accuracy of the database fills generated by the system. The criteria are limited to those that can be measured without access to anything more than the templates that the systems generated. Using these criteria, we attempt to assess the current state of the art, measure progress relative to previous evaluations, and compare the task performance of machines with that of humans.

The scoring software classifies each piece of extracted information into one of the following scoring categories: *correct, partial, incorrect, spurious, missing,* and *noncommittal.* Systems are penalized for having missed pertinent information, for having "hallucinated" more information than was actually pertinent, and for having otherwise extracted mismatching pieces of information. In order to reveal information about the circumstances affecting performance, the scoring software calculates scores at the following levels of granularity: for each slot in each template, for each object type in each template, and overall for each template; for each slot in the test set, for each object type in the test set, and overall for the test set.

Two sets of metrics were in force for MUC-5 [4]. The first set of metrics is based on the classification error rate and includes an overall metric (*error per response fill*) and three secondary, diagnostic metrics (*undergeneration, overgeneration, and substitution*). These secondary metrics correspond to the three penalty situations described above. The error per response fill metric and the secondary metrics are together referred to as the *error-based metrics.*

The second set of metrics measures the completeness (*recall*) and accuracy (*precision*) of the extracted information. These are supplemented by the undergeneration and overgeneration metrics mentioned above, which serve to isolate the system's shortfall in recall due to undergeneration and the system's shortfall in precision due to overgeneration. Recall and precision are combined into a weighted overall measure called the F-measure. Recall, precision, and F-measure are together referred to as the *recall-precision-based metrics.*

---

[2] "Core" refers to a core set of JV template slots: the **<template>content** slot, the **<tie-up-relationship> status, entity,** and **joint-venture** slots, the **<entity> name, aliases, location, nationality, type,** and **entity-relationship** slots, and the **<entity-relationship> entity1, entity2, rel-ent2-to-ent1,** and **status** slots.

[3] "Partial" means that all slots except those in the **<packaging>** object were part of the evaluation.

[4] The EJV test set for the MUC dry run contained fewer articles than the 18-month Tipster evaluation, due to restrictions on the right to use articles from some sources for MUC-5.

[5] The UMass/Hughes team was not yet under contract and did not participate in this evaluation.

[6] The UMass/Hughes team was tasked to work only in English (EJV, EME).

[7] All thirteen non-Tipster MUC-5 sites worked in just one domain; two worked in both languages, one worked in Japanese only, and ten worked in English only. See table 1.

The error-based metrics served as the official metrics for the MUC-5 evaluation, meaning essentially that any ranking of systems by overall performance would be done on the basis of error per response fill rather than F-measure. However, as it turned out, statistical significance tests showed that system rankings on the basis of error per response fill are very consistent with those made on the basis of F-measure (see discussion below and [3]). Both sets of metrics play important roles in the discussion found in this paper.

An appendix to this volume contains summary tallies and scores for each of the systems. The rightmost columns in the tables contain the scores for the error-based and recall-precision-based metrics; other columns contain the raw tallies. The rows in the top portion of the tables contain summary statistics for each slot and object; the rows at the bottom contain overall statistics. See the preface to the appendix and [4] for further information on reading the score reports.

Updates to the Test Design and Data

Each of the interim evaluations resulted in significant updates to the evaluation design. For the 12-month test, the evaluation software that had been used for MUC-4 was rewritten by SAIC to accomodate the object-oriented Tipster templates. Issues that were addressed in the interim between the 12-month and the 18-month tests include JV template formatting (especially in the Japanese template), performance metrics (probability of false alarm as alternative to precision, system-independent version of recall), object alignment by the evaluation software[8] (content-based as well as threshold-based alignment options, alignment optimization based on score rather than on number correct), and evaluation software support for human performance studies (scoring of one set of hand-coded templates versus another).

At the 18-month meeting, decisions were made regarding scoring for the 24-month/MUC-5[9] evaluation. The principal decision was to supplant recall and precision with a modified formulation of the error rate metric that had been in experimental usage for the 18-month test. The revised metric was named *error per response fill* because it is system-dependent (i.e., the denominator in the formula varies across systems according to the number of spurious fills generated, and it also varies because the answer keys allow for a somewhat variable number of expected fills). Error per response fill became the primary measure of performance. However, a less system-dependent error rate metric was also implemented; this metric was termed the *richness-normalized error*. These changes to the metrics necessitated significant reprogramming of the evaluation software. In addition, the decision was made to convert portions of the JV template from objects to complex slots[10], and this resulted in significant updates to the evaluation software, JV corpora, and JV documentation. Another decision resulting from this meeting was to ease the object alignment criteria, largely because of the difficulty of setting valid threshold values given the sparseness of the fills in many of the objects in the answer key.

The MUC-5 dry run was conducted after all these updates had been completed. Between the dry run and the final test two months later, further updates were made to the evaluation software, including a revised way of scoring two-part (complex) slots in the JV template. The new method gives separate scores to each part of the two-part fill, rather than giving one score to the complex fill as a whole. Another update was the implementation of a limited two-pass object alignment strategy, which results in slightly improved object alignments because more of the information on the interrelationships among entities is present when objects that reference the entities are aligned.

The intention had been to eliminate some evaluation criteria before the MUC-5 effort began in earnest in March, 1993; however, some of the decisions made at the 18-month meeting were tentative and, in the end, few simplifications were made at that time. The net result was that the number of performance measures has increased since MUC-4, and it is clear that there is still no clear answer as to the single most appropriate criterion to apply to assessing performance on an information extraction task. The good news is that the error per response fill and

---

[8]Object alignment as implemented for MUC-5 is discussed in the next section.

[9]Since the MUC-5 evaluation was the 24-month evaluation for the Tipster contractors, the evaluation will hereafter be referred to simply as the MUC-5 evaluation.

[10]This conversion affected three parts of the JV template: **ownership percent**, **product/service**, and **activity site**. After the conversion, each of these was represented in the template as a two-part slot rather than as an object with two slots.

the F-measure provide consistent views of the relative performance of systems, and therefore technology consumers may choose to use whichever set of metrics *they* feel is most appropriate for their purposes. All this experimentation resulted in other useful information as well about system-independent metrics, object alignment approaches, and template design, among other things.

Alignment and Scoring

System-generated (*response*) templates must be aligned with the answer-key (*key*) templates for scoring. Alignment takes place at all levels where there exist more than one response instance of a given kind and/or more than one key instance. These levels include the template level, the object level, and the slot-fill level. In each case, the intent is to find the alignment that will provide the best content match between the key and the response. At the object level, there is also the intent to determine whether a response object should be rejected for alignment purposes for failing to show *any* substantial degree of match with a key object.

Alignment at the template level is trivial; it is done on the basis of matching the **<template> doc nr** fills in the key and response. At the slot-fill level, when there are more than one key and/or response slot-fill for a given instance of a slot type, alignment is done on the basis of the degree of match between key and response fills. Slot-fill alignments that the alignment program can only guess at may be revised interactively during the scoring stage.

Alignment at the object level is the most complicated and controversial aspect of alignment. It takes place prior to scoring, and it is normally done fully automatically because to do it interactively would be so time-consuming as to be virtually impossible. The criteria for establishing whether an object ought to be allowed to align at all are defined in a file external to the alignment process. The criteria are defined to apply across all instances of a given object type. However, it is difficult to specify the criteria in this manner since many instances in the keys contain little fill on which to base a comparison.

Various object alignment schemes and minimal alignment criteria (also called the *minimal mapping requirements*) have been tried; for MUC-5, an alignment scheme called *threshold-based* was used, and the alignment criteria were loose. As used for MUC-5, this scheme allows nearly any matching fill in a given object type to enable an object alignment. The only exception concerns certain slots for which an overwhelming default fill exists, e.g., **<entity>type**. Such slots are ignored in the alignment process.

If there is no content match at all between a response object and a key object or if the only match is on a slot that is excluded from the threshold-based alignment criteria, the response object is marked as spurious. In such cases, the object's alignment status is termed *connected*, meaning that the object did not align but there existed a key object to which it could have mapped had it met the minimal alignment criteria. As a connected object under the official All-Objects scoring method (see [4] and preface to score-report appendix), response fills with no corresponding key fill are scored as spurious and those with a corresponding key fill (whether the fills themselves are a correct match or not) are scored as incorrect.

For a given object type in a template, there may be more than one possible alignment of object instances that meet the alignment criteria. Such objects are aligned on the basis of the degree of slot-fill match, as coarsely determined by the alignment program. The program determines an approximate error per response fill score, which will be overridden during the actual scoring process following alignment.

The alignment of objects in one pass results in suboptimal mappings of some object types, especially **<entity>** in the JV template, because advantage cannot be taken of useful information about the dependency between **<entity>** (or **<person>**) and **<entity-relationship>**. The solution implemented for MUC-5 was to align objects in two passes, with a few of the object types handled in both passes. However, despite the theoretical advantages of two-pass alignment, it is believed that the adopted solution results in only slightly improved object mappings over what can be done in a single pass. Two-pass alignment is only a partial solution to the problem, but the problem itself appears to be relatively minor.

MEASURING TASK DIFFICULTY

With each new MUC, the evaluators have challenged technology to deal with a broader variety of texts and to do more with them. One of the ways in which MUC-5 distinguishes itself from previous evaluations is in the increased task realism, which manifests itself in a greater variety of data extraction requirements, in the requirement for translation of extracted information into entries from standard reference sources (unabridged gazetteers, the Standard Industrial Code manual, etc.), and in a richer template structure. However, the most distinctive feature of Tipster Phase 1 for extraction is the requirement to handle more than one language and more than one domain. This requirement generated a strong push in the direction of language- and domain-independence, while the task realism generated a strong push for maximizing task coverage with minimum time and effort.

Major changes have been made to the evaluation design over the years, which complicates the issue of progress assessment. Not only have the metrics and scoring and alignment algorithms evolved and been replaced, but new extraction tasks have been defined [10]. The first two tasks were in the naval tactical domain, the next two were in the terrorist domain, and the Tipster/MUC-5 evaluation was conducted in the joint ventures and microelectronics domains. One could conceive of trying to compare the difficulty of these tasks in terms of human performance; however, at this point there exists reasonably sound performance data only for MUC-5 [11, 12]. One could also imagine trying to measure relative difficulty in some atheoretic or polytheoretic way in terms of the number of semantic patterns, inference rules, etc., required to carry out the task, but that idea is not a practical one.

In a preliminary attempt to compare the difficulty of different extraction tasks, quantitative criteria were developed in support of MUC-3 that enable comparison in terms of superficial features of the texts, template definition, and template fill rules [5]. Comparison of the complexity of the terrorist task with the naval task in light of these criteria shows at least an order-of-magnitude increase for several of the criteria. Once allowances are made for changes to the scoring methods and the earlier evaluation results are recomputed, it is clear for the results of the top systems in each evaluation that MUC-3 system performance represents significant progress for extraction systems as a group over the previous evaluation.

DIMENSION	FACTOR
Text corpus complexity	~1x
Text corpus dimensions	~3x
Template fill characteristics	~1-2x
Nature of task	~2x

Table 3. Difficulty of EJV task compared to terrorism task

The criteria can be adapted to allow rough estimation of the relative difficulty of the MUC-5 joint ventures and microelectronics tasks compared to the MUC-3/MUC-4 terrorism task. Most of the adaptations reflect the shift from a flat-format template to an object-oriented template. Table 3 summarizes the comparison, using EJV as the MUC-5 point of comparison.

The summarized data indicate that the EJV task is a somewhat more difficult task than the terrorism task along three of the four major dimensions. The dimensions measure difficulty in the following terms:

Text corpus complexity measures difficulty in terms of coverage of language features that may be encountered during testing. Measurement takes the following statistics from the training corpus into account:

- number of text types
- vocabulary size
- average sentence length
- average number of sentences per text

Text corpus dimensions measures difficulty in terms of the volume of material to be processed in order to achieve coverage and monitor system progress. Measurement takes the following statistics from the training corpus into account:

- number of texts

- number of sentences
- total number of words

Template fill characteristics measures difficulty in terms of features of the template structure and the amount of information to be extracted from a given test set. Measurement takes the following statistics from the training corpus into account[11]:

- number of object types[12]
- number of slots
- overall difficulty of slot types

This measurement also takes into account the following statistics from the MUC-5 test set[13]:

- percent nonrelevant texts
- average number of relevant events per relevant text
- average number of fills per slot

Nature of task measures difficulty in terms of the extraction task in general -- the elaborateness of the rules that the system must incorporate in order to conform to the template definition and fill rules, including relevance rules at the template, object, and slot level and the formatting specifications at the slot-fill level. Measurement takes into account the following statistics from the training corpus and MUC-5 test set:

- percent nonrelevant texts

This measurement also takes the following statistics drawn from the task documentation into account:

- number of pages of relevance rules
- number of pages of template definition and template fill rules

The numerical factor corresponding to each dimension in table 3 represents a rough average of the factors assigned to the component criteria identified above. Some of the assumptions inherent in this approach to assessing relative difficulty are that longer sentences will be processed less accurately than shorter ones, that relevant texts with a greater amount of relevant information present more opportunities for error, that a greater variety of extraction requirements makes a task harder, and that extraction is harder when it goes beyond categorization of information into a set fill.

The EJV task is harder by a factor of two on criteria such as the following:

- vocabulary size;
- average number of sentences per text;
- number of slots in the template;
- one of the types of slot (numeric/complex slots).

Among the ways in which EJV is *easier* than the terrorism task are the following:

- sentences in the EJV corpus are shorter on average (18 words versus 27 words);
- there are so few nonrelevant EJV texts that relevance filtering plays a negligible role (~10% nonrelevant versus ~50% nonrelevant);
- there is a sparser amount of information in the EJV templates (~1 filler per slot versus ~1.5 per slot).

The greatest difference between the EJV and terrorism tasks concerns the *text corpus dimensions*. This dimension, which treats the volume of text as a measure of difficulty, could be viewed as less of an issue now than it was for MUC-3. In fact, with the increasing popularity of statistical techniques, large amounts of training data are sometimes *required*. Nonetheless, the challenge of making effective use of text increases with the quantity of text, since a large amount of text implies a broad domain, and most kinds of domain knowledge cannot currently be captured using automated training methods.

---

[11] One other statistic that was used in comparing the naval and terrorism tasks, the number of template types, was not used in this comparison because the statistic is not pertinent to the way the Tipster templates are designed.

[12] In the MUC-4 template, there were no objects, but there were groupings of slots into those that contained data on the perpetrator of the terrorist act, the physical target of the terrorist act, the human target of the terrorist act, and on the terrorist act itself. These four slot groupings were referred to as *pseudo-objects*.

[13] Two other statistics that could be used if two object-oriented tasks were being compared--average number of objects per template and average number of slots per object--were not used in this comparison because there were no formal object types in the terrorist template.

The *percent nonrelevant texts* criterion, which figures in two of the dimensions, is based on the view that the more a system's performance would suffer as a consequence of ignoring the text filtering (document detection) subtask, the harder the task. The percentage of nonrelevant texts in EJV is so low (approximately 5% in the training corpus and 10% in the MUC-5 test sets) that a system can almost ignore the text filtering subtask without suffering a serious degradation in performance; the system can be optimized in favor of generating tie-ups even when it is not sure there is sufficient information in the text. This is not true of the terrorism task, where the percentage of nonrelevant and relevant texts is about equal. In conclusion, either a task such as EJV that places extremely little emphasis on text filtering or a task that places extremely high emphasis on text filtering is considered to be less difficult than one such as the terrorism task, which places significant emphasis both on text filtering and information extraction.

Within the context of the extraction subtask independent of the text filtering subtask, the more information there is to be extracted, the more difficult the task is judged to be. This is because richer texts present more opportunities to miss information and to confuse information about one reportable item with another.

The most difficult comparison to make concerns the *template fill characteristics*, because of the switch to the object-oriented template. Furthermore, the *overall difficulty of slot fill* criterion is itself composed of several features. It is based on the number and distribution of the various types of slots: set-fill slots with no more than twelve possible fills, set-fill slots with more than twelve possible fills (for MUC-5, these were slots that referenced the gazetteer), numeric/complex slots (which includes some normalized fills and, in the case of MUC-5, some two-part fills), string-fill slots (and normalized strings such as corporation names), and pointer-fill slots (in the case of MUC-4, these are slots that require cross-references). The more open-ended the extraction task, the harder it is judged to be. The EJV task is judged to be harder with regard to the numeric/complex slots in particular.

In summary, the generalization may be that the EJV task is harder than the terrorism task in terms of the template (number and nature of slots), the sheer volume of text (vocabulary size), and the discourse demands (number of sentences per text), but a little easier in terms of the shorter sentence length, lesser proportion of relevant information in relevant texts (number of fills per slot), and very small proportion of nonrelevant texts.

OVERALL RESULTS

The discussion of the MUC-5 evaluation results will be presented from various perspectives, using the metrics that are most appropriate in each case. This paper presents some general views on the results. Results for individual sites are summarized in the papers in this volume that were prepared by the evaluation participants.

Progress Assessment from MUC-4 to MUC-5 (EJV)

Since the F-measure was in force for both MUC-4 (as an official metric) and for MUC-5 (as an unofficial metric), a rough measure of progress can be obtained with that metric, using EJV as the representative MUC-5 task. The purpose of the comparison is to gauge whether the field of NLP as a whole has progressed in terms of overall performance achievable on extraction tasks. To that end, only the top-scoring systems are included in the comparison, namely those that were in one of the top two ranks statistically according to the F-measure.[14]

There were four systems in the top two ranks for MUC-4 (TST3 and TST4 test sets) [2] and three in the top two ranks for MUC-5 (EJV test set) [3]. These systems are GE, GE/CMU, UMass/Hughes, and SRI for MUC-4, and GE/CMU, BBN, and SRI for EJV MUC-5. The average F-measure score of the MUC-4 systems is 51.68; the average for the MUC-5 EJV systems is 47.12. If the one non-Tipster EJV system (SRI) is excluded from the EJV MUC-5 average, the average rises to 49.35.

The greater level of difficulty of the MUC-5 EJV task and the fact that the F-measure scores are close to being as high as the MUC-4 F-measure scores indicate that performance of top MUC-5 EJV systems is at least comparable to performance of top MUC-4 systems. It is important to remember that the Tipster systems were achieving that level of performance for MUC-5 on EJV while working also in the microelectronics domain and,

[14]The "P&R" F-measure value is used. This value weights precision and recall equally.

in most cases, also in Japanese. In that regard, it is notable that the GE/CMU system scored in the top rank in each language and domain pair; on the F-measure their scores were 52.75 for EJV, 60.07 for JJV, 49.18 for EME, and 56.31 for JME. It is also notable that SRI, which was a non-Tipster MUC-5 participant in both EJV and JJV, achieved F-measure scores of 42.67 and 44.21, respectively.

The fact that relative task difficulty can be assessed only roughly together with the fact that several MUC-5 sites worked on more than one task mean that too much importance should not be placed on comparison of scores between MUC-4 and MUC-5. However, whether or not difficulty factors and evaluation design changes are taken into account, there is at least one MUC-5 task on which performance can only be said to be outstanding, namely the JJV core-template task. Two systems achieved an F-measure score on the JJV core-template test in the 70-80 range -- 73.54 (corresponding to error per response fill of 39) for the GE/CMU Shogun system and 77.94 (error per response fill of 34) for the GE/CMU optional test run with the TEXTRACT system. Top performance on the EJV core-template test was about 20 points worse. The relatively high performance on the JJV core-template task may be indicative not only of the relative simplicity of the core-template task compared to the full-template task but also to the relative simplicity of the JJV texts compared to the EJV texts. (Some of these language differences are discussed further in a later section.) Nonetheless, taken on the task's own terms, these JJV scores reflect strong performance.

Comparison of Machine Performance with Human Performance

Application perspective. The F-measure is a weighted combination of recall and precision. Recall and precision give an indication of system performance relative to the application goals of extracting all and only the information that should be extracted. Despite the fact that humans are subject to human factors limitations that inhibit their performance, the performance limits of humans on an information extraction task represent a good target for automated systems as well, since the shortfall of human performance from perfection is due not only to human factors but also to other factors, such as deficiencies in the task definition. As reported in [11, 12], human performance and machine performance on 120 articles in the MUC-5 EME test set was measured. As part of the study, the performance of the four well-trained analysts and the top three MUC-5 systems (GE/CMU, BBN, and UManitoba) was compared.

The four human analysts were able to extract up to 79% of the information expected (recall metric), and of all the information they extracted, at best 82% of it was judged to be correct (precision metric). Performance of the top systems fell far below human performance; the three systems used in the comparison were able to extract up to 53% of the information expected, and of all the information they extracted, at best 57% of it was judged to be correct. In terms of performance shortfall, the machines fell 19-38 points of human performance on the recall measure and 18-31 points short of human performance on the precision measure.

Increasing system recall and precision by another 20 points or so may not seem to be a difficult task -- after all, since systems managed to obtain an F-measure score in the 70s on the JJV core-template test, why not also on the EME task? But it may not be easy to increase both recall and precision by that amount simultaneously on a relatively difficult task such as EME, since the metrics are in tension with each other. The harder a system tries to extract all the expected information (i.e., the more aggressively configured it is), the more likely it is to extract erroneous information. The tension is reduced if the texts are easier to interpret, as the JJV texts apparently are (see section below on handling two languages) and if the task is simpler, as the JJV core-template task undoubtedly is in comparison to the EME (full-template) task.

The overall recall and precision scores hide the fact that there were not only slots on which human performance was relatively strong but also slots on which human performance was relatively weak. A study reported in [11] measured the degree of difference between human and machine performance for frequently-filled slots in a portion of the EME test set. The author's general conclusion was that machines did comparatively well on slots that may lend themselves to keyword analysis and that are to be filled with a set-fill category from a relatively long list; examples include the **<layering> type** and **film** slots.

Speed. Another respect in which systems showed an advantage over humans is in terms of speed. On average, the time required for a human to fill a template (using software tools tailored for the Tipster tasks) was between 15 minutes (for an EME template) to over 60 minutes (for a JJV template). In contrast, timing information collected for the BBN PLUM system, the GE/CMU Shogun system, and the NMSU/Brandeis Diderot

system shows that the average time required to process an article in the EME test set was between 75.0 seconds (Shogun on a Sparc10 with 64 mb RAM) and 211.2 seconds (Diderot, which was not optimized for speed in English, on a Sparc2 with 32 mb RAM) and that the average time required to process an article in the JJV test set was between 39.0 seconds (Diderot on a Sparc2 with 2.32 mb RAM) and 140.8 seconds (PLUM on a Sparc10 with 128 mb RAM).

Predominant Classes of Error

The most frequent type of error committed by nearly all of the MUC-5 systems was to miss pertinent information. This class of error is captured by the undergeneration metric. The test results show that performance on this metric is a good indicator of performance on the overall metric of error per response fill. The effect of undergeneration in relation to the overgeneration and substitution metrics as well as to the error per response fill metric can be seen in figures 1 and 2, which graph the results of all MUC-5 systems for the EME and JME tests. From these graphs it is clear that undergeneration (UND) generally correlates with the overall metric of error per response fill (ERR), overgeneration (OVG) does not correlate with it, and substitution (SUB) correlates with it only to a limited extent.

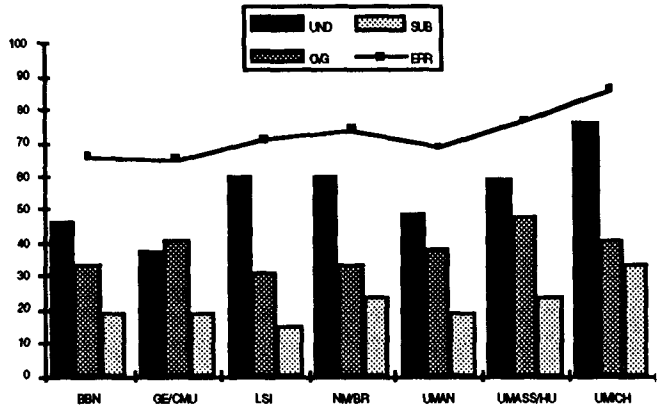

Figure 1. Classes of error and overall error per response fill for all EME MUC-5 systems

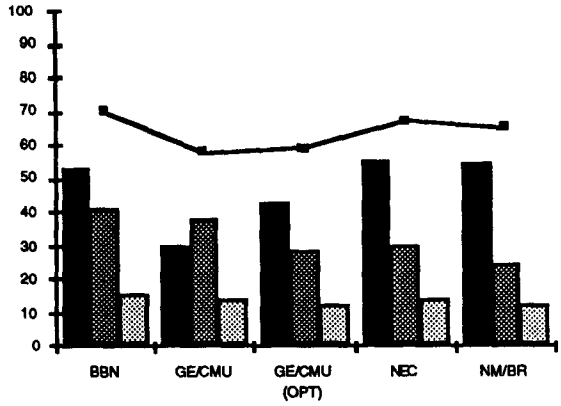

Figure 2. Classes of error and overall error per response fill for all JME MUC-5 systems

Substitution is a lesser source of error than undergeneration and overgeneration, lesser even than overgeneration. Examination of the template-fill specifications sheds light on these data. Some slots and objects in the JV and ME templates have essentially fixed number, requiring one fill or allowing zero or one fill; others have a highly variable number, some requiring one or more fills and some allowing zero or more fills. Thus, for the slots having a highly variable number of fills, there is no absolute bound on the number of fills a system could potentially spuriously generate. This means that overgeneration on those slots could be quite high. Substitution errors, on the other hand, are accrued only when there exists a pairing between a fill in the key and a fill in the response, and the response is judged to be incorrect. Thus, the substitution score has as its upper bound the number of fills in the key.

If the extraction of relatively little target information is indicative of poor overall performance, how and to what extent does the extraction of relatively *much* information -- good or bad -- correlate with overall performance? Are the aggressive systems just wildly guessing, or is their aggressiveness paying off form them on the overall metric? The data show that there is a correlation between generating lots of data and obtaining a relatively good (i.e., low) error per response fill score. This can be seen by computing the number of right and wrong fills generated by a system (this number is called the *actual* (ACT)) as a percentage of the total number of fills expected (termed the *possible* (POS)) and comparing that percentage with the overall error per response fill score.

In figure 3, the EME results are sorted by increasing error per response fill on the vertical axis. It is evident that the more fills generated by the system, the better its error per response fill score, even to the extent that the

number of fills generated by the GE/CMU system *exceeds* the number expected, i.e., the system clearly generated a high proportion of spurious fills (as figure 1 bears out). The only clear exception to the generalization is the UMichigan system, which had a relatively high error per response fill score despite having generated relatively many fillers (more than the Language Systems, Inc. (LSI) system or NMSU/Brandeis system). Figure 1 shows that the UMichigan system suffered from relatively high overgeneration as well as relatively high undergeneration.

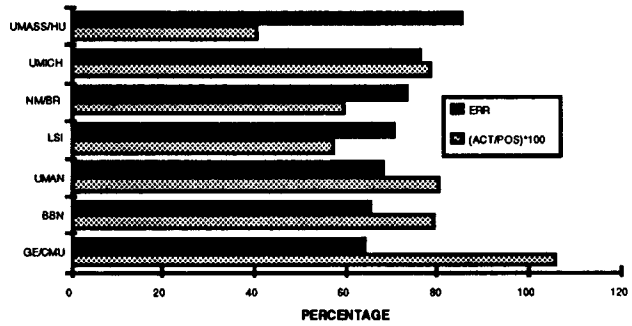

Figure 3. MUC-5 EME system aggressiveness in comparison with performance on the error per response fill metric

The results of the comparative performance study of machines (the GE/CMU, BBN, and UManitoba systems) and humans on part of the MUC-5 EME task show how far short of human performance the machines' performance fell. Well-trained humans are being compared with the best-performing MUC-5 systems. Since the MUC evaluations are designed to challenge research technology as well as to show a practical use of technology, it would probably be unreasonable to expect that any information extraction system participating in a MUC evaluation would perform at a level close to humans, and it is unlikely that any of the MUC-5 participants had comparability with humans as their primary development goal. Nonetheless, there may be evaluation data to help support speculation about how likely it would be that these systems could be developed to make up the shortfall.

Figure 1 shows that all EME systems other than GE/CMU incurred more errors as a result of missing information than as a result of committing other types of error, and figure 3 shows that generating more data was generally beneficial in terms of improving overall performance. The fact that the BBN and UManitoba systems' overall performance is very close to GE/CMU's -- in fact, the differences among the three are statistically insignificant [3] -- provides evidence that relatively good performance does not necessarily come at the expense of high overgeneration[15] and therefore that greater task coverage could make up for some of the shortfall from human performance. Further evidence of the room left for improvement of most, if not all, MUC-5 systems is found in the fact that there are slots which systems *never* filled in during the final test.[16] Even in the case of Tipster systems, these unattempted slots can account for a sizeable proportion of the total number of missed pieces of information.[17]

Measuring the Performance of Systems at Different Levels of Maturity

Scoring of unfilled slots. An object that is instantiated in the answer key may not be fully filled; the corresponding text may not provide information to fill some of the slots defined for that object type. Cases where

[15]This is not to fault the GE/CMU system for overgenerating. There are other systems with an equal or worse overgeneration score that come nowhere near matching the GE/CMU system in error per response fill. The GE/CMU system had undergeneration and overgeneration in close balance for the MUC-5 evaluation and evidently was optimized on both. Of the four language-pair systems they were required to field for MUC-5, three came out slightly better on balance on recall (which emphasizes minimizing undergeneration) and one, JJV, came out slightly better on precision (which emphasizes minimizing overgeneration).

[16]Count of unattempted slots (i.e., those where the system's "actual" equals zero) excludes those slots that were never filled in the key (i.e., those where the "possible" equals zero).

[17]For example, BBN's JJV system made no attempt to fill 17 of the slots in the JJV template, which accounts for 25% of the total missing, and their JME system made no attempt to fill 12 of the JME slots, accounting for 24% of the total missing; the UMass/Hughes system made no attempt to fill 13 of the EJV slots and 11 of the EME slots, and in each case this accounts for 15% of the total missing.

a template slot is correctly left unfilled by the system under evaluation are scored as *noncommittal* by the scoring software. Noncommittals are not included in the standard formulation of any of the performance measures. This is reasonable from a research perspective, if not from an applications perspective. The question comes down to whether systems normally leave a slot unfilled out of knowledge or whether they do so out of a lack of knowledge. Highly immature systems tend either to overgenerate to an extreme, leaving few slots unfilled, or to undergenerate to an extreme, leaving many slots unfilled. The latter type of immature system was very common at the MUC-5 evaluation and could have benefited unfairly from a metric that considers a noncommittal fill to be a correct fill, especially since there are many unfilled slots in the key templates.

The effect of scoring noncommittal fills as correct fills is to give an inflated estimate of performance, at least for the systems that undergenerate to a relatively large extent. It also has the potential effect of giving a distorted cross-system view, since very immature systems could end up being ranked higher than is intuitively sensible.[18]

The latter effect was not evident, however, for MUC-5, despite the relatively large number of unfilled slots in the answer keys (EJV compared to MUC-4). Apparently, the potential effect on the MUC-5 evaluation was eliminated through the object structure. Since the MUC-5 templates consist of objects that are aligned separately, the scoring impact of producing an object that fails to meet the minimal alignment criteria is limited to just that one object. Such an object, which contains an insufficient amount of correct fill to warrant alignment, is not given credit for any "correct" fills.[19] Thus, even though the object alignment criteria were loose for MUC-5, there were still objects that failed to align, and systems got no credit for any correct information that they may have contained.

For MUC-4, on the other hand, there was no object alignment, only template alignment, and the template alignment criteria were fairly strict. Thus, although no credit would be gained for correct fills in an unaligned template, the amount of credit that would be obtained for noncommittal fills in an *aligned* template would be fairly high on average, since the MUC-4 template is a larger structure than any of the objects in the MUC-5 template.

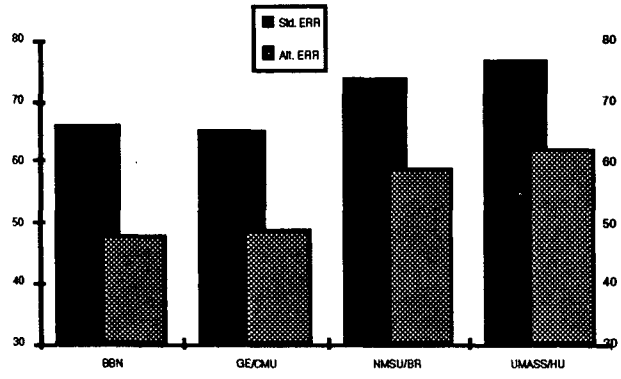

Figure 4. Tipster system MUC-5 EME scores for two formulations of error per response fill

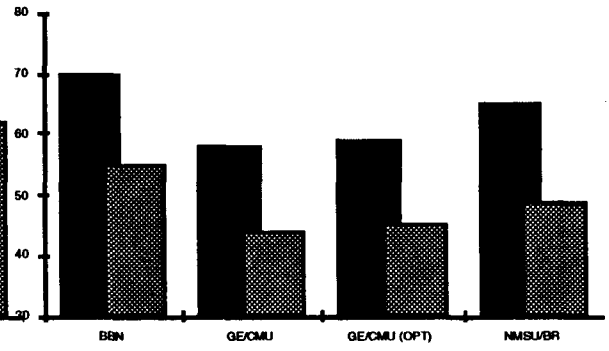

Figure 5. Tipster system MUC-5 JME scores for two formulations of error per response fill

Figures 4 and 5 provide examples of the difference the treatment of the noncommittal scoring category can make in the MUC-5 results. They show the error per response fill scores for the Tipster systems on EME and JME MUC-5 using two formulations of the metric: the standard formulation, which disregards noncommittal

[18]When applied to MUC-4 systems, the standard formulation of error per response fill results in no significant reranking of the 17 systems. But a formulation that includes noncommittals would result in rerankings of all 17 systems. The most radical changes would be for immature systems whose number of noncommittals grealy outweighs all other categories of response.

[19]Major differences between MUC-5 and MUC-4 in the alignment process do not play a role in this investigation of the scoring of noncommittal fills, since the investigation with respect to both MUC-5 and MUC-4 treated such fills as correct only in the scoring stage, not in the alignment stage. As far as scoring method goes, the two evaluations are not very different; both used the All-Objects method, which for MUC-4 was called All-Templates.

fills, and the alternative formulation, which treats noncommittal fills as correct. The alternative formulation and the standard formulation provide consistent cross-system views of performance; as discussed above, the alternative formulation does *not* distort the cross-system perspective on the results.

Viewed in terms of the impact on the actual scores, the difference between the two formulations ranges from 14 to 18 points.[20] As mentioned earlier, the alternative formulation inflates the scores of systems that greatly undergenerate. It is quite likely that such systems leave slots unfilled ignorantly more often than they do so knowingly. Nonetheless, actual performance of the systems may be estimated to lie somewhere between the two values, closer to the standard value for lesser developed systems and closer to the alternative value for more highly developed systems.

Richness-Normalized Error. The alternative error per response fill formulation described above may provide better insight than the standard formulation into the potential performance level of systems that miss relatively little of the pertinent information in the texts. Similarly, the richness-normalized error, either in its standard formulation or in an alternative formulation, may provide better insight than the error per response fill into the potential performance level of systems that generate relatively little spurious information, i.e., that have a relatively low overgeneration score. This metric views documents as streams of data of varying richness according to the number of fills in the key.[21]

The richness-normalized error metric is close to being a system-independent metric, meaning that the denominator disregards spurious responses because the number of such responses varies greatly from one system to the next.[22] Since overgeneration was a significant problem for virtually all MUC-5 systems, this measure tends to distort cross-system comparisons by treating systems relatively harshly that overgenerate to a relatively large extent. For this reason, this measure does not appear to offer a useful way of viewing the MUC-5 test results; however, it may be useful when the performance of systems under evaluation is uniformly higher.

Handling Two Languages

Four of the five sites that were evaluated in both Japanese and English (see table 1) performed at least as well in Japanese as in English. Averaged across *all* the MUC-5 systems, JME error per response fill is better than EME by eight points, and JJV is better than EJV by eleven points. Averaged across all the sites and the two domains, there is a ten-point difference between Japanese and English.[23] These findings are presented and analyzed in [7].

System performance differences between the two languages in the JV domain are attributed largely to differences between the EJV and JJV text corpora in terms of overall text structure and style. Analysis of the Japanese text characteristics and their impact on extraction performance is presented in [6]. The JJV corpus is more nearly homogeneous and the texts and sentences more pattern-like, which reduces the discourse demands and generally facilitates extraction.

In the ME domain, differences in scores between the two languages may be attributed largely to the fact that there was one-third less information to extract in JME than EME (average of 17 fills per template in JME, 25 fills per template in EME), including only about one microelectronics capability per template in JME as opposed

---

[20] For EJV and JJV, the difference is somewhat less, ranging from 9-13 points.

[21] Thus, data extraction is viewed as analogous to speech recognition. Just as in speech, where there are detectable and classifiable signals coming in, in data extraction there is extractable information coming in. The slot-fill count for a document is analagous to the word count for a stream of speech, and the slot fills in the key templates are analagous to the known words in the spoken sentences.

[22] However, it is not entirely system-independent. A small amount of system dependence remains because of variability in the key templates, which capture some textual ambiguity by representing alternative correct answers, which may include an alternative *number* of slot fills or objects in a particular instance [4]. This situation may arise in speech as well, where hearers disagree on which words and how many words were uttered.

[23] These statistics are based on the results for all MUC-5 sites. Consequently, English JV and ME averages are low because of the number of relatively underdeveloped, non-Tipster systems that were evaluated in English only. If the statistics are limited to those sites that worked in both languages (five JV, three ME), there is still a five-point difference between Japanese and English (six-point difference for JV and four-point difference for ME).

to about two per template in EME. Thus, the problem of object splitting and merging (discourse-related template effects) is lesser in JME. Discussion in [7] goes into more depth on this subject, showing interesting performance differences between EME and JME in relation to the key templates that contain multiple **<process>** objects.

Handling Two Domains

The four Tipster sites (five systems, including the GE/CMU optional JJV and JME optional test runs using the CMU TEXTRACT system) were evaluated in both extraction domains. Although they had more time to work on JV than ME, their system performance was comparable across domains. Overall error per response fill scores for the UMass/Hughes system are the same for EJV and EME; the BBN system performed a little better on ME than JV (two points difference in both languages); the GE/CMU system scored worse on ME than JV (four points difference in both languages); and results for the NMSU/Brandeis system are mixed -- better on ME than JV for English (five points difference) and a little worse for Japanese (two points difference). The biggest difference was shown on the GE/CMU optional test run (nine points worse on JME than JJV).

It would appear that the comparable results achieved by most of the systems are attributable primarily to factors that kept JV performance down. Parts of the JV template underwent many changes, which may have caused sites to do less development on those parts. Some sites may have also skipped parts that represented a very small proportion of the overall task in terms of number of fills in the training corpus keys, especially skipping deeply embedded slots and/or objects. In addition, the fact that testing was conducted on a core portion of the template as well as on the full template may have caused sites to focus less development effort on the non-core portions of the template.

The net effect of these factors is that the sites essentially reduced the task to a manageable size and as a consequence, incurred errors by missing relatively more information in JV than ME. Although this generalization holds for most of the MUC-5 systems, among the Tipster systems it does not apply to the GE/CMU Shogun English system, the GE/CMU TEXTRACT (optional) Japanese system, or the NMSU/Brandeis Japanese system. Statistics on the average degree of task reduction by the MUC-5 sites in each language-domain pair can be found in [7].

RESULTS FOR LIMITED JV TASK

Overall Performance

MUC-5 English and Japanese joint ventures testing was conducted in two configurations. In one configuration, the entire template was scored; in the other, only the core portion of the template was scored (see footnotes to table 2). Figures 6-9 graph error per response fill together with the diagnostic secondary metrics of undergeneration, overgeneration, and substitution for the Tipster systems for each of the two configurations. Across the EJV systems, the error per response fill scores on the core-template test range between seven and nine percentage points better (lower) than on the full-template test; for the JJV systems, the error per response fill scores on the core-template test range between fifteen and sixteen points lower than on the full-template test.

The source of most of the difference in error per response fill is in the number of missed fills, which is reflected in better undergeneration scores on the core-template test; the range across Tipster systems is 6-15 points lower for EJV and 11-24 points lower for JJV. The only other sizeable differences (i.e., differences of more than five points) are the overgeneration score for the GE/CMU EJV and JJV systems (nine points lower on the core-template test for EJV and seven points lower for JJV) and both the overgeneration and substitution scores for the GE/CMU optional JJV run using the CMU TEXTRACT system (overgeneration nine points lower on the core-template test and substitution seven points lower). Thus, for all systems except GE/CMU's, the only score among the secondary metrics that differs considerably between the two test configurations is the undergeneration score.

The difference in scores on the two configurations is more marked for Japanese than for English, with the best error per response fill scores posted for the whole evaluation by the GE/CMU Shogun system and the GE/CMU optional test run with the TEXTRACT system on the JJV core-template test (scores of 39 and 34, respectively). On the EJV and JJV full-template tests, most of the error per response fill scores are in the 50-70

range. As a point of reference, the error per response fill score of 61 posted by the GE/CMU system on the EJV full-template test corresponds to a recall of 57 and precision of 49 (F-measure of 52.75).

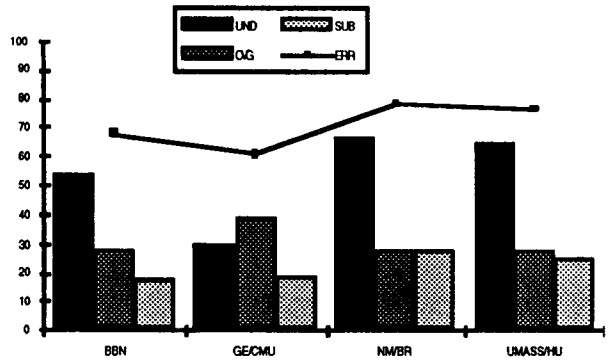

Figure 6. Tipster system scores for the EJV full-template test

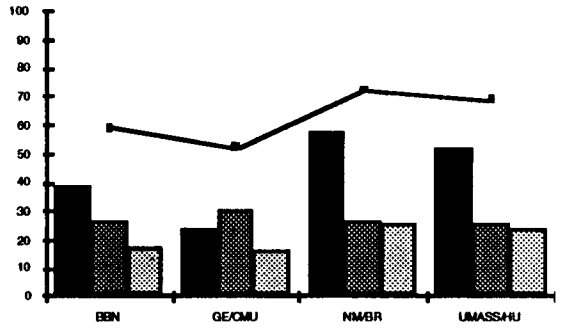

Figure 7. Tipster system scores for the EJV core-template test

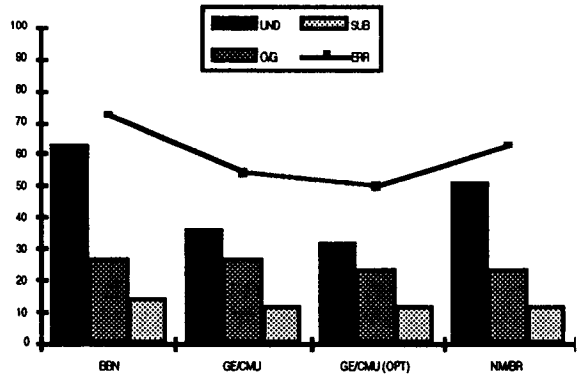

Figure 8. Tipster system scores for the JJV full-template test

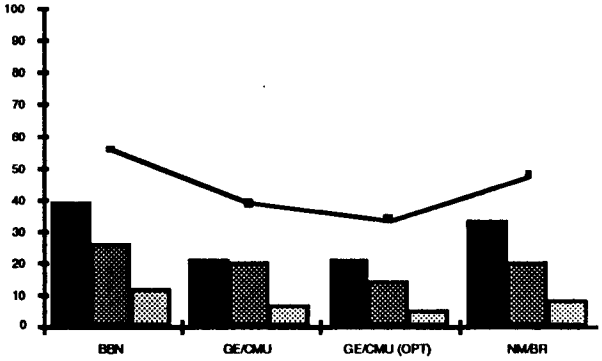

Figure 9. Tipster system scores for the JJV core-template test

Slot-Level Performance

The JV core template includes fourteen slots, one-third as many slots as the full template; yet for the EJV MUC-5 full-template test, the slot fills from the core slots account for nearly two-thirds (around 63%) of the total slot fills. This distribution reflects the fact that the core-template slots cover some of the less idiosyncratic portions of the task. Since the MUC-5 test set is fairly representative of data seen in the training corpus, it is not surprising that participants would have dedicated more development effort to the core slots in the template and would have been able to leverage previous work that is applicable across a range of tasks.

Therefore, it is not surprising that scores on the core slots are relatively good compared to other slots in the template. At least one of the four Tipster EJV systems had an error per response fill score of less than or equal to 50 on six of the 43 scored slots[24]; five out of the six slots are in the core part of the template. At least one of the systems scored between 51 and 75 on twenty other slots; nine of the twenty are in the core part of the template. Scores over 75 were obtained for many non-core slots but not for any core slots. Statistics for the Tipster EJV system that scored best on each slot and for the average across Tipster EJV systems are summarized in table 4.

[24]The **<rate>eta** slot is excluded from the total slot count, since there were no fills for it in the key for the EJV MUC-5 test.

ERR Range	#Slots: Best Slot Score	#Slots: Average Slot Score
0-25	0	0
26-50	6 (5)	1 (1)
51-75	20 (9)	12 (9)
76-99	14	27 (4)
100	3	3

Table 4. Tipster EJV performance on slots (best and average score) by range in error per response fill. Numbers in parentheses are for core-template slots.

The fact that performance on the core slots is relatively good is evident if the template slots are divided into categories roughly according to their type: pointer, set fill, string fill, numeric fill, geographic place-name fill, temporal fill, two-part (complex) fill. The core template contains slots of the following types: pointer, set fill, string fill, and geographic place-name fill. For each of the Tipster EJV systems it is generally the case that performance on the core slots of a given type is better than performance of any other slots of that type. Thus, for example, performance by each of the Tipster EJV systems on the four set-fill slots in the core set (**<entity>type**, **<tie-up-relationship>status**, **<entity-relationship> status**, **<entity-relationship>rel-ent2-to-ent1** is better than performance on any of the four set-fill slots that are not in the core set (**<industry>type**, **<facility>type**, **<person> position**, **<revenue>type**).

There are three minor exceptions, which affect only the NMSU/Brandeis and UMass/Hughes systems. Two of the exceptions show performance on a non-core set-fill slot slightly better (two points) than **<entity-relationship>rel-ent2-to-ent1.** The third exception is that the UMass/Hughes system performed four points worse on **<entity>aliases**, a core string-fill slot, than on **<person>name**, which is a non-core string-fill slot.

However, in addition to these minor exceptions, there are two core slots that represent *major* exceptions that affect all four of the systems: **<tie-up-relationship>joint-venture** and **<entity-relationship>entity2**. These are both pointer slots to an **<entity>** object. For each system, there is at least one non-core pointer slot (and as many as five) that the system scored better on than on than these two core slots. Furthermore, there is a gap between the scores for these two core pointer slots and the scores for the other core pointer slots of at least nine points (and as many as seventeen).

The **joint-venture** and **entity2** slots have similarities that indicate why performance on them is not as good as on the other core pointer slots: they both require making two-way role distinctions among entities found in the texts, and they both capture the less frequent of the two entity roles. In the case of the **<tie-up-relationship>** object, both the **joint-venture** and the **entity** slots point to an **<entity>** object, but the **joint-venture** slot is meant to be filled only when a tie-up results in the formation of a joint venture company, which is often not the case. In the case of the **<entity-relationship>** object, both the **entity1** and **entity2** slots point to an **<entity>** object, but the **entity2** slot is meant to be filled only if a relationship exists other than partnership, which is the most common type of relationship. The lower scores on **joint-venture** and **entity2** are therefore attributed in part to the relative difficulty of identifying specific roles of entities.

The restricted use of the **joint-venture** and **entity2** slots is reflected in the template definition: **joint-venture** and **entity2** are constrained to contain either zero or one filler while the **entity** and **entity1** slots must contain at least one filler and may contain two or more. The system must decide not only what to fill the **joint-venture** and **entity2** slots with but also whether to fill them at all. Thus, the system is likely to fill them only if it has found clear evidence in order to avoid generating spurious data, and this can result in the opposing type of performance problem, namely missing relevant information.

Apart from the **joint-venture** and **entity2** slots, the only core slots that appear to have suffered relatively poor performance for all four systems compared to other core slots are the **<entity> location** and **<entity>nationality** slots, two of the three geographic place-name slots in the template. Although the systems scored better on these two place-name slots than on the non-core one, **<facility>location**, the fact that all systems appeared to have relative difficulty with those two core slots is notable, as it may reflect a practical difficulty of selecting the correct entry for an ambiguous place name from the large English gazetteer as

well as the linguistic difficulty of determining whether a mention of a place in association with an entity reflects the entity's location or its nationality.

However, there is a problem with attributing relatively low performance of the four core slots under discussion solely to the difficulty of determining the correct role of an entity or of a geographic place-name. The problem is that the lower performance may also be partially explained by the fact that those slots are less frequently filled than any of the other core slots in the full-template test.[25] All other core slots account for at least 3% each of the fills in the full-template test, with six core slots in the 3-4% range and five slots in the 5-10% range. Thus, even among the core slots, it can be expected that development efforts were not focused equally on all slots and that lower performance on some core slots may be a consequence not only of their relative difficulty but also of their lesser impact on the total evaluation.

SUMMARY

The evaluations conducted during Phase 1 of the Tipster extraction program have measured the completeness and accuracy of systems and have used an examination of the role of missing, spurious and otherwise erroneous output as a means of diagnosing the state of the art. Viewed as a set of performance benchmarks for the state of the art in information extraction, the MUC-5 evaluation yielded EJV results that are at least as good as the MUC-4 level of performance. This comparison takes into account some of the measurable differences in difficulty between the EJV task and the MUC-3 and MUC-4 terrorism task.

However, even a superficial comparison of task difficulty is hard to make because of the change from the flat-format design of the earlier MUC templates to the object-oriented design of the MUC-5 templates. Comparison is also made difficult by the many changes that have been made to the alignment and scoring processes and to the performance metrics. Therefore, it is more useful to view performance of the MUC-5 systems on their own terms rather than in comparison to previous MUC evaluations.

From this independent vantage point, MUC-5 yielded very impressive results for some systems on some tasks. Error per response fill scores as low as 34 (GE/CMU optional test run using the CMU TEXTRACT system) and 39 (GE/CMU Shogun system) were obtained on the JJV core-template test. The only other error per response fill scores in the 30-40 range were achieved by *humans*, who were tested on the EME task; however, machine performance on that EME test was only half as good as human performance. Thus, while the JJV core-template test results show that machine performance on a constrained test can be quite high, the EME results show that a similar level of machine performance on a more extensive task could not be achieved, at least not in the relatively short development period allowed for ME.

Not only do results such as those cited for the JJV core-template test show how well some approaches to information extraction work for some tasks, they also show how manageable languages other than English can be. A cross-language comparison of results showed fairly consistent advantage in favor of Japanese over English. Comparison of results across domains does not show an advantage in favor of one domain over the other, and it is quite likely that differences in the nature of the texts, the nature and evolution of the extraction tasks, and the amount of time allowed for development all had an impact on the results.

The quantity and variety of material on which systems were trained and tested presented challenges far beyond those posed by earlier MUC evaluations. The scope of the evaluations was broad enough to cause most MUC-5 sites to skip parts of the extraction task, especially types of information that appear relatively rarely in the corpus. Since no type of information is weighted in the scoring more heavily than any other, the biases that exist in the evaluation reflect the distribution of relevant information in the text corpus and result in a natural emphasis on handling the most frequently-occurring slot-filling tasks. These tasks turn out to be the ones that are less idiosyncratic and therefore more important to the development of generally useful technology.

---

[25] However, these four core slots are more frequently filled than many of the *non-core* slots. Of the 30 non-core slots, 24 account for less than 3% each of the total fills (13 account for less than 1% each, and 11 account for 1-2% each); only six of the non-core slots account for a sizeable proportion of the total fills (four account for 3-4% each, and only two account for 5-10% each).

Examination of the slot-level results in the appendix to this volume shows which systems are filling which slots and how aggressively they are generating fills. For those slots where a system is generating a substantial number of fills, analysis at the level of the individual templates and corresponding texts would provide insight into the particular circumstances under which the system extracted correct or incorrect information. In other words, the quantitative performance measures may yield information on aspects of performance that deserve further analysis, but a deeper investigation needs to include examination of the actual fills and the actual texts. The discussion in this paper of slot-level performance on the JV core-template task does not go as far as that; the discussion is based only on frequency of slot fill and on the slot definitions. Some of the deeper analysis can be carried out only by the authors of the systems. Such an analysis would relate the circumstances under which correct or incorrect system behavior was seen with the strengths and weaknessses of particular algorithms and modules of the system.

ACKNOWLEDGEMENTS

The author would like to acknowledge the extensive support throughout Tipster Phase 1 from Nancy Chinchor and Gary Dungca at SAIC. She would also like to acknowledge the active participation of the ARPA/SISTO sponsors, Thomas Crystal and George Doddington, and the long-suffering assistance of the MUC-5 program committee, which included Nancy Chinchor of SAIC, representatives of the Tipster contractors -- Sean Boisen of BBN, Jim Cowie of NMSU, Joe McCarthy of UMass, and Lisa Rau of GE --, representatives of other MUC-5 sites -- Ralph Grishman of NYU, Jerry Hobbs of SRI, and Carl Weir of Unisys --, and representatives of the DoD Tipster management -- Lynn Carlson, Mary Ellen Okurowski, and Boyan Onyshkevych.

REFERENCES

[1] Carlson, L., et al., Corpora and Data Preparation, in this volume.

[2] Chinchor, N., The Statistical Significance of the MUC-4 Results, in *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, June 1992, San Mateo: Morgan Kaufmann, pp. 30-50.

[3] Chinchor, N., The Statistical Significance of the MUC-5 Results, in this volume.

[4] Chinchor, N. and Sundheim, B., MUC-5 Evaluation Metrics, in this volume.

[5] Hirschman, L., Comparing MUCK-II and MUC-3: Assessing the Difficulty of Different Tasks, in *Proceedings of the Third Message Understanding Conference (MUC-3)*, May 1991, San Mateo: Morgan Kaufmann, pp. 25-30.

[6], Maiorano, S., An Analysis of the Joint Venture Japanese Text Prototype and Its Effect On System Performance, in *Proceedings from the TIPSTER Text Program, Phase One*, September 1993, San Mateo: Morgan Kaufmann (to appear).

[7] Okurowski, M.E., Domain and Language Evaluation Results, in this volume.

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

[9] Science Applications International Corporation, *Tipster/MUC-5 Scoring System User's Manual*, version 4.3, August 1993.

[10] Sundheim, B. and Chinchor, N., Survey of the Message Understanding Conferences, in *Proceedings of the ARPA Human Language Technology Workshop*, March 1993, San Mateo: Morgan Kaufmann.

[11] Will, C., Comparing Human and Machine Performance for Natural Language Information Extraction: Results from the Tipster Text Evaluation, in *Proceedings from the TIPSTER Text Program, Phase One*, September 1993, San Mateo: Morgan Kaufmann (to appear).

[12] Will, C., Comparing Human and Machine Performance for Natural Language Information Extraction: Results for English Microelectronics from the MUC-5 evaluation, in this volume.