BBN PLUM:
MUC-4 Test Results and Analysis

Ralph Weischedel, Damaris Ayuso, Sean Boisen,
Heidi Fox, Herbert Gish, Robert Ingria,

BBN Systems and Technologies
10 Moulton St.
Cambridge, MA 02138
weischedel@bbn.com

GOALS

Our mid-term to long-term goals in data extraction from text for the next one to three years are to achieve much **greater portability to new languages and new domains, greater robustness, and greater scalability.** The novel aspect to our approach is the use of learning algorithms and probabilistic models to learn the domain-specific and language-specific knowledge necessary for a new domain and new language. Learning algorithms should contribute to scalability by making it feasible to deal with domains where it would be infeasible to invest sufficient human effort to bring a system up. Probabilistic models can contribute to robustness by allowing for words, constructions, and forms not anticipated ahead of time and by looking for the most likely interpretation in context.

We began this research agenda approximately two years ago. During the last twelve months, we have focused much of our effort on porting our data extraction system (PLUM) to a new language (Japanese) and to two new domains. During the next twelve months, we anticipate porting PLUM to two or three additional domains.

For any group to participate in MUC is a significant investment. To be consistent with our mid-term and long-term goals, we imposed the following constraints on ourselves in participating in MUC-4:

- We would focus our effort on semi-automatically acquired knowledge.
- We would minimize effort on handcrafted knowledge, and most generally.
- We would minimize MUC-specific effort.

Though the three self-imposed constraints meant our overall scores on the objective evaluation were not as high as if we had focused on handtuning and handcrafting the knowledge bases, MUC-4 became a vehicle for evaluating our progress on the long-term goals.

MEASURING SUCCESS IN ACHIEVING OUR SHORT-TERM GOALS

PLUM had demonstrated quite high recall in MUC-3 and scored among the top systems. We chose to focus on the following goals in MUC-4:

- Increasing precision and reducing overgeneration, without hurting recall.
- Demonstrating a broad range of tradeoff in recall and precision.

Goal 1: Increasing precision and reducing overgeneration, without hurting recall. As the graph in Figure 1 shows, we doubled our precision in MUC-4 (compared to MUC-3) and reduced our overgeneration by roughly one third. The overall impact was to increase PLUM's F-Measure by 50%. Naturally one would ideally base this measurement on the new test sets for MUC-4 (TST3, TST4); however, between MUC-3 and MUC-4 both the definition of the templates to be produced and the evaluation function changed dramatically, so that there was no easy way to run the MUC-3 version of PLUM on TST3 and TST4 to produce results comparable to that of the MUC-4 version of the system. However, since the Government had converted our MUC-3 TST2 templates to the MUC-4 format, and since we had never examined the corpus of messages or the answer key to TST2, we could easily use it as a basis for comparison.

Goal 2: Demonstrating a broad range of tradeoff in recall and precision. As Figure 2 illustrates, the user can select from a broad range of system performance, emphasizing either recall or precision to various degrees. No system had displayed such a span favoring recall versus favoring precision in MUC-3. Only one other system, GE's, demonstrated a broad range; at a cost of 17 points of recall, GE's system could achieve an increase of roughly 8 points of precision. For PLUM, the tradeoff of recall for precision was far more balanced.

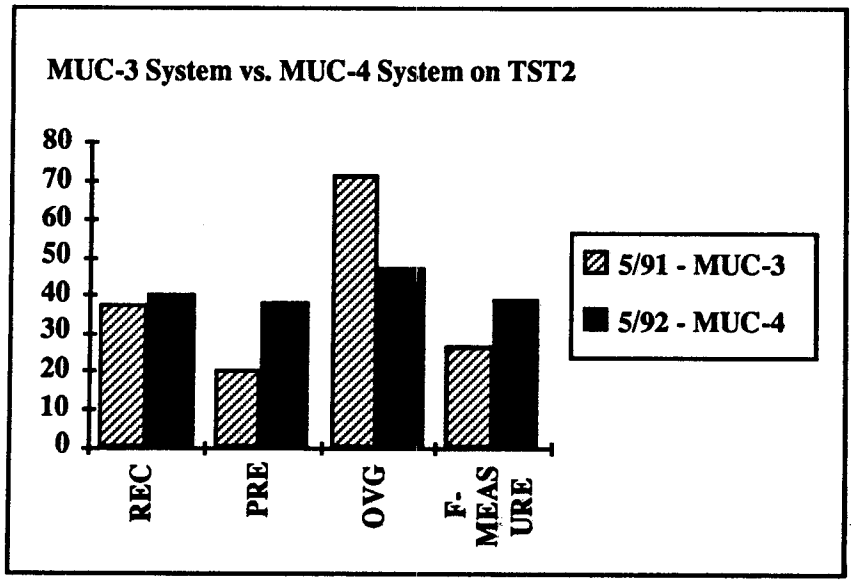

Figure 1: PLUM's precision was doubled; overgeneration was cut by 1/3; overall performance (F-Measure) increased 50%; all without hurting recall.

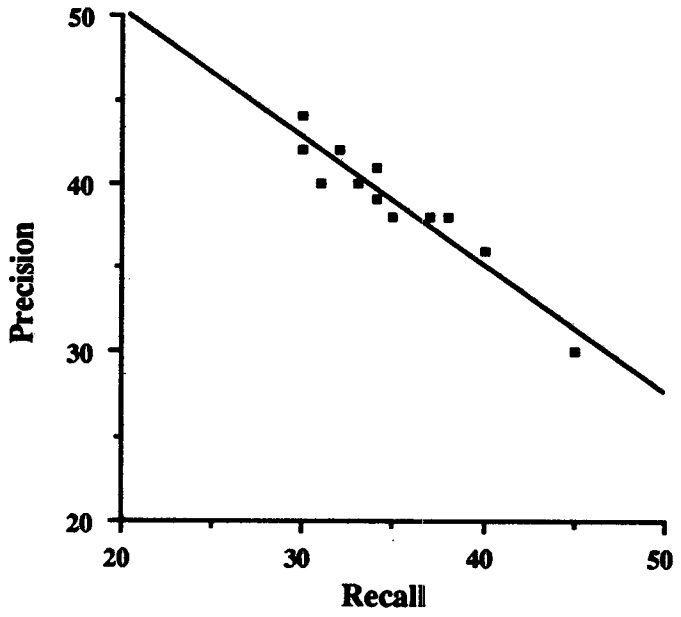

Figure 2: By varying parameters, a wide range of recall and precision can be obtained.

Two independent parameters primarily contributed to this: a discrete parameter controlling how aggressively or conservatively two descriptions are fused into a single view of the same event, and a continuously variable threshold on a classification algorithm predicting whether a paragraph is relevant or irrelevant (with respect to reporting any terrorist incident). Together, these two parameters offer a user the ability to turn a knob to emphasize recall or precision based on their application preference.

KEY SYSTEM FEATURES

Two design features stand out in our minds: partial understanding and statistical language modeling. By *partial understanding* we mean that the parser and grammar are designed to find analyses for a non-overlapping sequence of fragments. When cases of permanent, predictable ambiguity arise, such as a prepositional phrase that can be attached in multiple ways, or most conjoined phrases, the parser finishes the analysis of the current fragment, and begins the analysis of a new fragment. Therefore, the entities mentioned and some relations between them are processed in every sentence, whether syntactically ill-formed, complex, novel, or straightforward. Furthermore, this parsing is done using essentially domain-independent syntactic information. The semantic interpreter and the rest of the system in turn do not assume having complete understanding.

The second key feature is the use of *statistical algorithms to guide processing*. Determining the part of speech of highly ambiguous words is done by well-known Markov modeling techniques. To improve the recognition of Latin American names, we employed a statistically derived five-gram (five letter) model of words of Spanish origin and a similar five-gram model of English words. This model was integrated into the part-of-speech tagger.

Another usage of statistical algorithms was a statistical induction algorithm to learn case frames for verbs from examples. This saved substantial effort compared to building the case frames by hand. The algorithm and empirical results are described in [3].

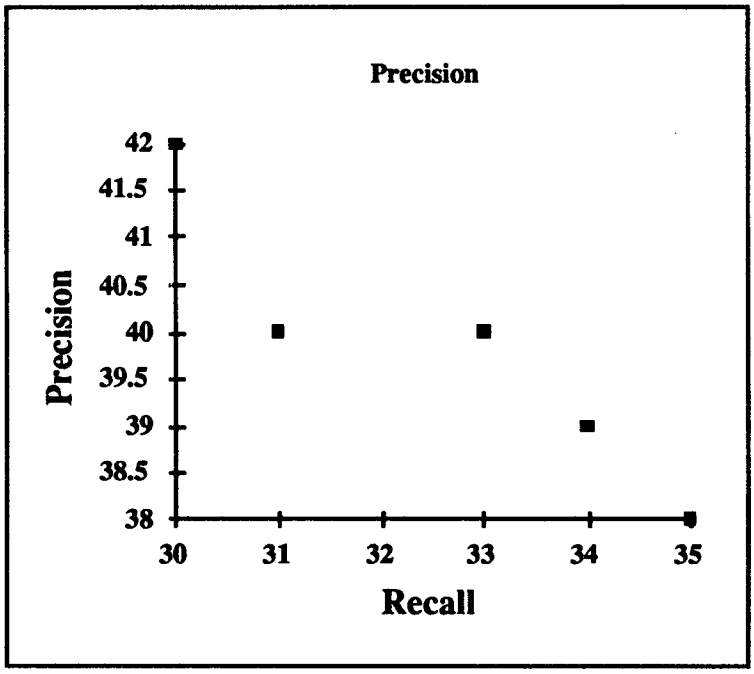

Figure 3: Impact of Paragraph classifier on recall and precision in the ALL TEMPLATES row.

The statistical methods mentioned above were already available and used in the MUC-3 version of PLUM. A new statistical algorithm employed in MUC-4 is a classification algorithm that automatically learns features to discriminate among classes. Given a list of relevant paragraphs and a list of irrelevant ones (made available by New Mexico State University), we employed a chi square measurement to determine word stems (though other features could be used as well) whose presence or absence in text is significantly correlated with the text being relevant (or irrelevant). Given that ordering, the user must select how many features to use. At runtime, the classifier sums the logarithm of the odds that the paragraph is relevant given the presence of the features. If the sum exceeds a user-specified threshold, the paragraph is considered relevant. If the classifier predicts that the paragraph is relevant, then events found in the paragraph can be used to generate templates; if not, terrorist events that would otherwise have been produced from that paragraph are blocked. The performance of the overall system, given various thresholds of the text classifier, is shown in Figure 3.

A more detailed description of the system components, their individual outputs, and their knowledge bases is presented in Ayuso et al., [1]. We expect the particular implementations to change and improve substantially during the next three years of research and development.

RESULTS

Appendix G lists detailed test scores. A number of systems performed better on TST4 than on TST3, and some performed significantly worse on TST4 than TST3. The results on the two test sets were so disparate for PLUM that we decided to look into the causes of the abnormally low recall of PLUM on TST3. As table 1 shows, the following properties of TST3 stand in stark contrast with TST4, TST1, and the 1300 message development corpus:

- The percentage of relevant articles was abnormally high for TST3.
- The density of templates produced, i.e., the number of templates per 100 messages was abnormally high, roughly 50% higher than the 1300 message development set, TST1, and TST4.
- The density of human targets produced, i.e., the number of human target slot fills per 100 messages, was abnormally high, roughly double the number in TST1, TST4, or the development corpus.

	Relevant Messages	Irrelevant Messages	Marginal Messages	Required Templates per 100 Messages	Optional Templates per 100 Messages	Human Targets per 100 Messages
TST3	65%	31%	4%	119	17	162
TST4	48%	45%	7%	73	5	84
DEV 0001-1300	42%	50%	8%	71	11	93
TST1	55%	50%	5%	88	7	84
TST2	57%	35%	8%	100	28	151

Table 1: Based on this comparison of development messages and four test sets, TST4 is a more representative of the MUC-4 domain than TST3.

Taken together, the first two observations above suggest the following:

Systems tuned to overgenerate (i.e., produce a high percentage of templates, what Hirschman labels the "lazy merge problem" in this volume) should perform significantly better on TST3 than TST4.

The observations above suggest, in part, why PLUM's recall for TST3 was abnormally low:

- PLUM's merging algorithm is under user-control, and was set to avoid overgeneration, therefore yielding worse performance on TST3 than on TST4.
- A large number of human targets exercise two known weaknesses of the MUC-4 version of PLUM: (1) known, temporary grammar problems and (2) a challenge for discourse processing to be able to collect targets across sentences.
- A bug in the official scoring program was encountered for TST3, but not for TST4. If this bug is corrected, we estimate it would improve PLUM's scores by at least one point in recall, at least one point in precision, and at least two points in overgeneration. (That clearly is not sufficient to fully account for the discrepancy in performance on TST3 and TST4.)

One other point confirming the normalcy of TST4, contrasted with the abnormal characteristics of TST3, can be seen in PLUM's performance under various settings. Prior to the test, we ran PLUM with numerous parameter settings on TST1, TST2, and one set of 100 messages from the development set. This predicted the setting that would maximize the F-Measure, or come indistinguishably close to the maximum F-measure. That prediction proved correct (consistent) with TST4, but was 2 points under the maximum actually achieved for TST3 via one of our optional runs.

Table 2: A summary of scores on TST3 and TST4.

Required Run on TST3 (settings favor precision)

SLOT	REC	PRE	OVG
MATCHED/MISSING	30	69	10
MATCHED/SPURIOUS	51	44	43
MATCHED ONLY	51	69	10
ALL TEMPLATES	**30**	**44**	**43**
SET FILLS ONLY	33	71	14
STRING FILLS ONLY	23	64	12
TEXT FILTERING	83	87	13
	P&R	2P&R	P&2R
F-MEASURES	35.68	40.24	32.04

Optional Run on TST3 (settings favor recall)

SLOT	REC	PRE	OVG
MATCHED/MISSING	49	72	9
MATCHED/SPURIOUS	55	28	64
MATCHED ONLY	55	72	9
ALL TEMPLATES	**49**	**28**	**64**
SET FILLS ONLY	52	75	9
STRING FILLS ONLY	43	70	15
TEXT FILTERING	90	67	51
	P&R	2P&R	P&2R
F-MEASURES	35.64	30.62	42.61

Optional Run on TST3 (settings maximize F-measure)

SLOT	REC	PRE	OVG
MATCHED/MISSING	38	69	10
MATCHED/SPURIOUS	52	38	50
MATCHED ONLY	52	69	10
ALL TEMPLATES	**38**	**38**	**50**
SET FILLS ONLY	40	70	12
STRING FILLS ONLY	30	64	15
TEXT FILTERING	98	80	20
	P&R	2 P&R	P&2R
F-MEASURES	38.00	38.00	38.00

TST3 and TST4 Combined (settings favor precision)

SLOT	REC	PRE	OVG
MATCHED/MISSING	35%	71%	9%
MATCHED/SPURIOUS	52%	44%	44%
MATCHED ONLY	52%	71%	9%
ALL TEMPLATES	35%	44%	44%
SET FILLS ONLY	37%	73%	11%
STRING FILLS ONLY	29%	68%	12%
TEXT FILTERING	83%	81%	19%
	P&R	2 P&R	P&2R
F-MEASURES	38.74	41.76	36.13

Required Run on TST4 (settings favor precision)

SLOT	REC	PRE	OVG
MATCHED/MISSING	40	72	8
MATCHED/SPURIOUS	53	42	47
MATCHED ONLY	53	72	8
ALL TEMPLATES	**40**	**42**	**47**
SET FILLS ONLY	42	75	9
STRING FILLS ONLY	36	72	12
TEXT FILTERING	82	71	29
	P&R	2 P&R	P&2R
F-MEASURES	40.98	41.58	40.38

Table 2 summarizes PLUM's performance on TST3 where precision is maximized (the required run), where recall is maximized, and where F is maximized. It also lists the required run for TST4. In addition, since TST3 and TST4 were so disparate in character, we computed the score of PLUM if TST3 and TST4 together constituted the test.

EFFORT SPENT

We estimate that 4 person months specific to MUC-4 went into our effort. These were spent approximately as follows: domain-dependent lexical additions, 0.5 person months; grammar, 0.5 person months; semantic rules, 0.75 person months; discourse, 1.0 person months; backend, 0.75 person months; and overhead (evaluation, fulfilling requirements, etc.), 0.5 person months.

TRAINING DATA AND TECHNIQUES

The 1300 messages of the development corpus were used at various levels as training data. PLUM was run over all 1300 messages to detect, debug, and correct any causes of system breaks. The perpetrator organization slot for

all 1300 messages was used to quickly add names to the domain-dependent lexicon. After running our part-of-speech tagger (POST) over the development corpus, the statistical algorithm for predicting words of Spanish origin was run over the list of previously unknown words. Those predicted as Spanish in origin were then reviewed manually to add Spanish names to the lexicon.

A subset of the development set was used more intensively as training data. Approximately 95,000 words of text (about 20% of the development corpus) was tagged by the University of Pennsylvania as to part of speech and labelled as to syntactic structure as part of the DARPA-funded TREEBANK project. The bracketed text first provided us with a frequency-ranked list of head verbs, head nouns, and nominal compounds. For each of these we added a pointer to the domain model element that is the most specific super-concept containing all things denoted by the verb, noun, or nominal compound. As mentioned earlier, the TREEBANK data was then used with the lexical relation to the domain model to hypothesize case frames for verbs. The automatically hypothesized verb case frames were then reviewed manually and added to the lexicon. This is detailed in [3].

The 100 messages of TST1 and TST2 were used as blind test sets to measure our progress at least once a week. Throughout, we only looked at the summary output from the scoring procedure, rather than adding to the lexicon or debugging the system based on particular messages.

The training mentioned above had already been used in preparing for MUC-3, with the obvious exception that TST2 was not available in preparation for MUC-3. What we added in MUC-4 was training regarding the relevance/irrelevance of paragraphs. We tried training at the article level; however, the fact that an article could be mostly irrelevant except for a single paragraph mentioning a terrorist incident made the training much less effective than training based on labelling individual paragraphs as irrelevant.

CONCLUSIONS

Successes. Though the structure of PLUM did not change radically between MUC-3 and MUC-4, the one new component, a statistically based text classification algorithm, was quite successful. It was trained fully automatically. Once the system is trained on sets of text representing the various classes, such as relevant paragraphs versus irrelevant paragraphs for the MUC domain, the user need do only two things: select a cutoff for words to be used for each class in the log probability model and set a continuous variable which serves as the threshold for inclusion in a given class. The result was a significant, continuous tradeoff in recall versus precision.

A second new success in our experience this year was the heuristic for when to merge (fuse) two descriptions into a single event representation. Many of the heuristics do not require domain-specific knowledge. The heuristic was not knowledge-intensive, yet significantly reduced overgeneration, while increasing precision.

These two heuristics together enabled us to double precision and reduce overgeneration by one third, thus effecting an overall improvement in performance (F-measure) by 50% compared to the MUC-3 version of PLUM.

What Limited Success. Having an investment in the terrorist domain and FBIS corpus inhibited significant changes that we might otherwise have made, such as (1) moving away from a purely deterministic parser (i.e., a beam search of width one) to a more general probabilistically controlled beam search, (2) moving from the present semantic interpreter to a more declarative one, and (3) replacing the MUC backend with a general purpose one driven by knowledge bases derived in part from training data.

Improvements Desired. Improving syntactic coverage is a priority. Increased coverage normally leads to greater perceived ambiguity in the system; we hope to counter that through probabilistic models. A second priority is improving coverage of the discourse component. The template generator today is based on handcrafted rules of thumb. Within the next year we hope to develop and test an acquisition algorithm that would acquire most of the rules from examples in a new domain. Lastly, though the classification algorithm was a pleasant success, we believe an even more accurate classifier is possible.

Lessons Learned. There are several lessons we believe we learned this year:

1. **Automatic training and acquisition of knowledge bases can yield relatively good performance at reduced labor.** Suppose one plots F-measure on TST3 and TST4 together against total effort in MUC-3 and MUC-4 together. PLUM achieved high performance with very little labor (7 person-months in MUC-4 and MUC-3 combined). Of the top eight sites in MUC-4, compared to our effort, one group put in at least 50% more effort than we did; all other groups put in roughly 2-6 times the effort in MUC-3 and MUC-4 combined.

2. **Substantial tradeoff in recall and precision is achievable**, in particular nearly even tradeoff of recall for precision is achievable over a broad range.
3. **User control of the recall/precision tradeoff is attainable via a continuous variable**, e.g, via a knob the user can turn to prefer recall or to prefer precision in varying degrees.
4. **A test set of 100 messages seems too small to accurately assess system performance.** Scores for three systems decreased dramatically in moving from TST3 to TST4; of those, two were in the top eight systems. By contrast, scores for five systems increased dramatically in moving from TST3 to TST4. Since the discrepancy between TST3 and TST4 was great for half of the systems, and since those systems divided almost evenly on which test set gave higher performance, it is clear that a set of 100 messages, unless carefully chosen to balance characteristics of the test set, is too small.
5. **Unfortunately, given the statistics in Table 1, TST4 seems more representative than TST3.** With hindsight it seems clear that combining the results of TST3 and TST4 would have given a useful measure.[1]

ACKNOWLEDGMENTS

The work reported here was supported in part by the Defense Advanced Research Projects Agency and was monitored by the Rome Air Development Center under Contract No. F30602-91-C-0051. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the United States Government.

REFERENCES

[1] Ayuso, D.M., Boisen, S., Fox, H., Ingria, R., and Weischedel, R. "BBN: Description of the PLUM System as Used for MUC-4". *MUC-4 Proceedings*, 1992.

[2] Weischedel, R., Ayuso, D.M., Bobrow, R., Boisen, S., Ingria, R., and Palmucci, J., Partial Parsing, A Report on Work in Progress, *Proceedings of the Fourth DARPA Workshop on Speech and Natural Language*, 1991a.

[3] Weischedel, R., Meteer, M., and Schwartz, Applications of Statistical Language Modelling to Natural Language Processing, unpublished manuscript, 1991b.

---

1 One can reasonably combine the scores by adding the scoring entries for the columns POS (possible), ACT (actual), COR (correct), PAR (partial), MIS (missing), and SPU (spurious), then using the definitions of recall, precision, overgeneration, and F on the totals.