TIPSTER SHOGUN SYSTEM (JOINT GE-CMU): MUC-4 TEST RESULTS AND ANALYSIS [1]

George Krupka and Paul Jacobs
Artificial Intelligence Laboratory
GE Research and Development
Schenectady, NY 12301 USA
E-mail: rau@crd.ge.com
Phone: (518) 387 - 5059
Michael Mauldin and Todd Kaufmann
Carnegie Mellon University
Ira Sider
Military and Data Systems Operation
GE Aerospace

Abstract

This paper reports on the joint GE-CMU Tipster SHOGUN *customization effort for MUC-4, and analyzes the results of the TST3 and TST4 runs in comparison with the GE system.*

INTRODUCTION

We report on the joint GE-CMU results from the MUC-4 conference and provide an analysis of system performance. In its maiden test, the joint SHOGUN system performed very well. On the positive side, the system achieved very good overall results. On the negative side, because the system was barely ready in time for MUC-3, it was difficult to implement and test any significant modifications.

RESULTS

Our overall results on both TST3 and TST4 were very good in relation to other systems. Figure 1 summarizes our results on these tests. On both TST3 and TST4, the GE-CMU system was 9-10 recall points behind the GE system and about 4 F-measure points behind. The entire difference is due to the difference between the GE TRUMP parser, which had been developed for text applications and thoroughly tested in MUC-3, and the CMU generalized LR parser, which was developed for machine translation and has just begun to be tested on this sort of task.

In addition to these core results, the Figure 2 summarizes our performance on the adjunct test.

GE - GE-CMU SCORE COMPARISON

While the performance of the GE-CMU system was not as good as that of the GE system, we view these results in a very positive light. First, the system with the CMU parser was reasonably close to the GE system, and it was rapidly catching up at the end of the preparations for MUC-4 (The GE-CMU system improved about 30 recall points between the interim test in March and the final test in May). Second, the integrated system proved that software modules and fundamental results could be shared across sites,

[1]This research was sponsored (in part) by the Defense Advanced Research Project Agency (DOD) and other government agencies. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Project Agency or the US Government.

	TST3				TST4			
	REC	PRE	OVG	F	REC	PRE	OVG	F
Matched Only	66	72	13		69	68	18	
All Templates	49	55	33	51.8	53	53	37	53
GE AT	58	54	36	55.9	62	53	39	57.15

Figure 1: GE-CMU MUC-4 TST3 and TST4 Results

	TST-3			
	REC	PRE	OVG	F
1MT	60	58	29	58.9
1ST	57	35	63	43.4
2MT	34	50	37	40.5
NST	45	84	9	58.6

Figure 2: GE-CMU MUC-4 TST3 Adjunct Results

systems and methodologies, with some effort. The success of both the parser integration and the MUC test was thus encouraging for our TIPSTER effort as a whole.

EFFORT

We spent overall approximately 2 person-months of effort on the GE-CMU system, compared with 10 person-months on the GE system. Half of this effort was from CMU, where participants had not been through MUC before and there was thus a "learning curve" to get used to the test and testing procedures. The amount of development represented here is thus very small, and went almost entirely into improving the robustness and recovery of the parser.

The entire difference in scores is due to the difference between the two parsers, since the rest of the system components are the same. Successful parses covered 34% less of the input (We failed to get an accurate count of successful parses because of a bug in the calculation) in the LR parser, and unsuccessful parses produced 50% more fragments. Our rough analysis is that about half of the recall difference between the two systems is due to more parser failures in the GE-CMU system, and the remaining half is due to recovery problems. Although the same recovery strategies were used in both cases, the behavior of the parsers on failure was substantially different, so more work is needed on LR parser recovery.

Some problems we have noted that cause parser failure are: inability to cope with spurious and missing punctuation, as well as nested comma clauses, catastrophic failure on missing or spurious determiners, and problems with long sentences where we set a time limit.

Problems with recovery include a difference between the way the two systems handle "chain rules" in the grammar, a bug that prevented the LR system from building complete noun phrases on failure, and the lack of certain phrase reductions on end of sentence (the LR parser would often leave a long trail of fragments at the end of sentence, instead of failing early and trying to recover).

Most of these problems are relatively minor, but they take time to fix. Our system had *no* real parser recovery mechanism until a few weeks before MUC-4.

In addition to producing somewhat lower scores, the system with the LR parser was 7-9 times slower than the GE system using TRUMP. This difference was due to the fact that the LR parser was designed to follow many more parse paths, putting a large burden on the semantic interpreter to sort out the best interpretation from a huge "forest" of parses. We believe that this gap can be quickly closed with the installation of a new control module for the parser.

RETROSPECTIVE ON THE TASK AND RESULTS

The GE-CMU system in MUC-4 broke new ground in resource sharing. To our knowledge, two substantial natural language systems have *never* been successfully integrated, and the *way* the two systems were combined is a bit remarkable. We converted GE's grammar and lexicon to work with CMU's parser, used CMU's parser compiler on the GE grammar, and developed a separable data structure for parser recovery and control that could work with two vastly different parsers. This integration was so complete that, in at least one case, a bug turned up in the LR system's infant recovery module that led to a bug fix to *both* parsers.

The result is that grammars and lexicons developed for one parser can be used with another, control strategies can be tested with either parser, and the two parsers can be interchanged and compared at the "flip of a switch" within the context of the MUC or TIPSTER systems. This suggests that sharing of resources in a test like MUC may be considerably more practical than we had once believed.

LESSONS LEARNED

The MUC task gave the SHOGUN system a good test drive, providing a testbed for efficiently working the kinks out of the combined system. This benefit came not only from having a real task, but, surprisingly, from being able to compare the results of two interchangeable modules. If *both* parsers produced the wrong result, we would follow one line of response, while we would follow a completely different path if only *one* of the two parsers failed. Since much of the effort in MUC is determining what went wrong where, the interchangeable parsers turned out to be a useful diagnostic aid.

The amount of effort spent on pulling together the SHOGUN system for MUC, barely two person-months, proved that one system can really benefit from the work that goes into another. This is a mixed blessing, because we did pay double the price in the work required to fill templates, run tests, and run the scoring program on the two systems (two people did nothing but run tests and score!) The overhead of testing means that we wouldn't want to be doing this too often, but it seems a worthwhile investment for getting a relative analysis of two parsers.

SUMMARY

The SHOGUN system performed better than we had hoped on TST3 and TST4, proving not only that we had successfully integrated and adapted our new system, but also that strategies and entire modules can be shared beyond a particular system or site. As this was not our main goal when we set out, we are particularly emphatic about the positive prospects of comparing different approaches within a complete system.