SRI INTERNATIONAL RESULTS FEBRUARY 1992 ATIS BENCHMARK TEST

Douglas E. Appelt, Eric Jackson

SRI International
Menlo Park, CA 94025

ABSTRACT

We describe the results that SRI International achieved on the February 1992 ATIS Speech and Natural Language System Test. The basic architecture of the system is described, including a set of parameters capable of altering the system's behavior and processing strategy. We report on several experiments that were run on the February test set to evaluate several processing strategies for both natural-language only and full spoken-language system tests.

1. INTRODUCTION

This paper reports on the results of running SRI International's spoken-language system on the DARPA-sponsored February 1992 test. The system's natural-language processing has been parameterized in several ways to achieve different behaviors. In addition to running our system with what we believed at the time of the test to be the optimal parameter settings to produce our official results, we have conducted some experiments by running the system with a variety of parameter settings. The results of these experiments shed some light on the trade-offs among various SLS and natural-language processing strategies, and provide some interesting data for evaluating the evaluation methodology itself.

2. SYSTEM DESCRIPTION

The SLS system used for the February evaluation is an integration of the SRI DECIPHER speech recognition [1,4,5] system with the SRI TRAVELOGUE natural-language processing system. The integration between these two systems is currently accomplished by a simple serial interface: the best accoustic hypothesis is processed by the NL system to produce the answer to the query.

The DECIPHER System

DECIPHER is a speaker-independent continuous-speech speech recognition system based on tied-mixture Hidden Markov Model (HMM) models. It uses six features, three being vectors (cepstra, delta-cepstra, and delta-delta-cepstra) and three scalars (energy, delta-energy, and delta-delta-energy). These features are computed from a filter bank that is derived via an FFT and high-pass filtered (RASTA filtered) in the log-spectral-energy domain. DECIPHER models pronunciation variability through word networks generated by linguistic rules then pruned probabilistically. There are cross-word acoustic and phonological models. Parallel recognizers were implemented and trained separately on male and female speech. The DECIPHER-ATIS system uses a backed-off bigram language model to reduce the perplexity of the input speech.

The acoustic models were trained on all available ATIS spontaneous and read data (excluding 809 sentences used for system development that include 362 October 1991 dry run sentences and 447 MADCOW sentences). The backed-off bigram language model was trained on the available ATIS spontaneous speech data. This included 14,779 sentences (approximately 150,000 words). The recognition lexicon consisted of all words spoken in all available spontaneous ATIS data. There are also lexical entries for breaths and silence. No catch-all rejection model was used for out-of-vocabulary items. The vocabulary size is 1385 words.

The TRAVELOGUE System

The TRAVELOGUE system consists of a template-matching sentence-analysis mechanism [3] coupled with a context-handling mechanism and a database query generation component.

The template matcher operates by producing templates from the input sentence which then get translated into database queries. The two main components of a template are the template type, which generally corresponds to a relation in the underlying database, and a set of filled slots, which represent constraints present in the query. A template for the sentence "Show me the nonstop flights from Boston" might be of the type "flight" and have an origin slot filled with "Boston" and a stops slot filled with "0." In addition to these components, a template contains an illocutionary force marker (e.g., "show," "how many," "yes/no"), and a list of explicitly requested fields from the relation associated with the

template type. There are 20 different template types and 110 distinct slots.

The template matcher determines the type of template by looking for certain key nouns or key phrases in the sentence. It incorporates a simple noun phrase grammar that allows it to identify phrases containing key nouns. The presence of a key noun in certain contexts (e.g., in a noun phrase preceded by a word like "show") will more strongly trigger the associated template type than an isolated occurrence of that key noun. Conjunctions of noun phrases containing key nouns produce templates with multiple template types.

Slots are filled by matching regular-expression patterns against the input string. For example, "from" followed by an airport or city name may fill the origin slot of the flight template. To find fillers for slots, the template matcher makes use of a lexicon of names and codes, each associated with the appropriate sort, and special grammars for recognizing numbers, dates, and times. For each template type with some key noun or key phrase present in the sentence, the system tries to find the best "slot covering" of the sentence it can. That is, it tries to find the sequence of slot-filling patterns that matches the sentence and consumes as many words as possible. Two constraints are (1) slot filling phrases may not overlap, and (2) no slot may be filled twice with different values. The system incorporates a schematic mapping of the domain, which contains the information as to how entities are related, and allows the system to determine what slots are possible for each template.

In the next stage, the system chooses a single template from the set of candidate templates that have been constructed. It chooses on the basis of several factors, including the type of key that triggered the template and the number of words consumed in filling slots. A template score is then computed for the chosen template, reflecting the proportion of words in the sentence that are considered to be consumed. Words that fill slots or help slots get filled count, as well as function words and certain other words (such as "please") that are ignored for the purposes of scoring. If the template does not score above a threshold, the system chooses not to risk answering the query. The threshold can be varied depending on how much risk of a wrong answer can be tolerated. For evaluation we have found a threshold of about 0.85 to be optimal, while for data collection we use a lower threshold, typically 0.5.

The template matcher incorporates special mechanisms to handle certain types of false starts and complex conjunctions. These phenomena cannot be handled well in a straightforward, unaugmented, template-matching approach.

The template matcher was developed on all the annotated MADCOW data available as of January 1, 1992. In addition, a 3,000-sentence subset of the MADCOW data was annotated with the correct template for each utterance. The template production of the system could be quickly evaluated on these sentences. As of January 1992, the system's performance on this corpus was above 90%.

When a template is produced, the context-handling mechanism of TRAVELOGUE is invoked to determine whether the template for the current sentence should be modified or expanded based on the current state of the dialogue. The system employs a variety of context handling rules, each of which is justified by a plan-based model of dialogue structure similar to that of Grosz and Sidner [2]. The basic model tracks the context of a dialogue by assuming the user is following a plan that involves knowing which database entities satisfy a set of constraints that he or she has in mind when the session commences, because the user has the goal of formulating a travel plan (as opposed to other purposes for which such a database would be useful).

The context mechanism inherits constraints expressed by previous queries in a scenario as long as accumulating these constraints is consistent with knowing a single set of constraints applicable to a single travel plan. Knowing whether this set of constraints is consistent with the overall plan is accomplished by comparing the new slots to a context priority-lattice that establishes a partial order of dependencies among various template slots. Changes in higher-level constraints cause lower-level constraints to be discarded. This general mechanism is supplemented with a mechanism for handling deictic references and references to particular database entities that have appeared in answers to previous questions.

When a template including contextually inherited slots is produced, the TRAVELOGUE produces, optimizes, and runs a PROLOG database query, generating the final answer.

3. OFFICIAL RESULTS

In the February 1992 DARPA ATIS benchmark tests, SRI achieved the following results: In the ATIS speech recognition evaluation, SRI achieved a word recognition error rate of 11.0% and a sentence recognition error rate of 48.7% over all sentences on the test corpus. In the ATIS natural-language-only test, SRI achieved a weighted error rate of 31.1%, with 533 queries answered correctly, 60 incorrectly, and 94 given no answer. In the ATIS spoken-language systems evaluation, SRI achieved

a weighted error rate of 45.4%, with 444 queries answered correctly, 69 incorrectly, and 174 queries given no answer.

We performed an error analysis on the NL-only evaluation results. We examined all the queries that we did not answer or for which we were scored wrong, and tried to ascertain the cause.

Of the sentences that were either incorrect or unanswered, 46% can be attributed to the failure of the template matcher to generate a correct template. Of these failures, 80% could be remedied within the current framework while 20% would require a substantially different approach, such as a parser and grammar that together could provide more structural information about a sentence. We estimate that 12% of the errors were due to the database query generation component, and 18% were due to failures of the context mechanism to identify the correct context. The remaining errors are attributed to the system declining to answer questions when it determined that its uncertainty about the context was too great.

These figures were derived in a highly subjective fashion, but, nevertheless, we feel they give a roughly accurate picture. For a majority of the utterances that caused trouble for the template-generating component, it is clear that adding a new phrase or new slot could solve the problem. The conclusion we draw from this is that a template-matching approach can be highly successful on a domain of about the same complexity as ATIS. How well this type of approach would scale up to a significantly larger domain remains uncertain.

4. ADDITIONAL EXPERIMENTS

We have implemented several parameters that control the behavior of the system. One parameter is the template-matcher score cutoff.

We recognized that if a system failed to respond correctly to a query, it might give incorrect answers to a number of subseqent context-dependent queries, even though the subseqent sentences were processed correctly, given everything the system can determine about the state of the dialogue. Therefore, we have included several parameters that regulate the generation of responses in situations in which, for one reason or another, the state of the context is in doubt.

One such parameter is a cumulative template-score cutoff. We reasoned that if the system answers a series of questions, each of which receives an acceptable, although less than perfect, template score, eventually a point is reached in which the system is so uncertain about the correctness of the accumulated contextual information, that it should, for evaluation purposes, stop answering questions until a query is encountered that definitely sets a new top-level context. This point is detected by multiplying template scores until the cumulative product drops below the level indicated by the cumulative cutoff parameter. Our official results were produced by using values of 0.85 and 0.82 for the template score cutoff and cumulative score cutoff, respectively.

Another parameter controls the choice of one of three possible ways of dealing with the failure to produce an answer for a query. When the system fails to answer a query, it could refuse to answer any further queries until one is found that sets a new top-level context. Although this would be a ridiculous way for a system to behave when interacting with a real user, some preliminary investigation led us to believe that such a strategy was indeed optimal for the evaluation; this is the strategy used to generate our official results. Another possible strategy, which we dub "always answer," is to have the system answer every question in the last previously known context, regardless of how many intermediate queries fail to produce answers. Finally, we have a "usually answer" mode, in which queries are always evaluated in the most recently determined context, unless there is some feature of the query that indicates explicit dependency on a question that was not answered (such as a pronoun or demonstrative reference that could rely on an unanswered query for its resolution).

We ran experiments on our system for the following configurations of parameters on both NL and SLS data. These runs were made by changing only the parameters discussed above, without attempting to influence the behavior of the system in any other way:

1. **Relaxed Cutoff.** We set the template score cutoff to 0.82, and the cumulative cutoff to 0.70. Some of our earlier experiments suggested that these values were optimal for processing speech recognizer output. (Because of an oversight, they were not used in the official test).

2. **Low Scoring Template Strategy.** This strategy sets the template score cutoff and cumulative cutoff to be 0.01. This allows very low scoring templates to be considered as analyses for a sentence. The conservative strategy of not answering questions after failure to produce any template at all until the next context-resetting sentence was still followed.

3. **Maximum Recall Strategy.** This strategy combines the Low-Scoring Template Strategy with the Always Answer strategy. It seeks to maximize recall

by always answering a query whenever any analysis at all is possible. Naturally, precision suffers, because of the increased chance that some of the poorly rated analyses will be wrong.

4. **Maximum Precision Strategy.** We attempted to maximize the system's precision score by setting the template score cutoff and the cumulative cutoff to be 0.99. This strategy causes the system to respond only to templates with perfect scores and to stop answering in context whenever any uncertainty about a template exists. Naturally, because some correct templates will be discarded, recall suffers.

5. **Always Answer Strategy.** The "always answer" context-handling strategy was adopted, keeping the template score cutoff the same as in the official run.

6. **Usually Answer Strategy.** The "usually answer" context-handling strategy was adopted, keeping the template score cutoff the same as in the official run.

5. RESULTS OF EXPERIMENTS

The results we observed for the experiments described in the previous section (as well as our official results on the evaluation) were as follows, ordered by increasing weighted error:

For NL only:

Parameter Settings	Right	Wrong	No Ans	Wtd. Error
Always Answer	554	72	61	29.84
Usually Answer	538	60	89	30.42
Relaxed Cutoff	537	62	88	30.86
Official Results	533	60	94	31.05
Low-Score Template	558	90	39	31.88
Maximum Recall	565	98	24	32.02
Maximum Precision	480	38	169	35.66

For SLS:

Parameter Settings	Right	Wrong	No Ans	Wtd. Error
Always Answer	457	75	155	44.40
Relaxed Cutoff	447	69	171	44.98
Usually Answer	445	69	173	45.27
Official Results	444	69	174	45.40
Low-Score Template	455	86	146	46.29
Maximum Recall	460	93	134	46.58
Maximum Precision	423	62	202	47.45

As can be seen, the predicted parameter settings for Maximum Recall and Maximum Precision did result in the desired recall-precision tradeoff, although neither of these strategies produced the best results as measured by weighted error. It is also interesting to note that, with the exception of the tests for Relaxed Cutoff and Usually Answer configurations (which were in any case very close), the ordering of the results as measured by weighted error was the same for both NL and SLS tests.

6. SLS EVALUATION WITH BBN RECOGNIZER OUTPUT

Because the preliminary results of the February 1992 ATIS benchmark tests suggested that the SRI TRAVELOGUE NL system and the BBN BYBLOS speech-recognition system had both performed particularly well, SRI and BBN collaborated on an experiment to see how well a combined system would have performed on the benchmark test, using the output of BYBLOS as the input to TRAVELOGUE. We took the BYBLOS output from the official February 1992 ATIS SPREC test and ran it through TRAVELOGUE, configured exactly as it was for the official February 1992 ATIS SLS test. So, although this was not submitted as official February 1992 ATIS SLS test output, it is comparable in every respect to the official results obtained by BBN and SRI. The resulting combination produced 482 correct answers, 69 wrong answers, and 136 without answers, for a weighted error of 39.88%.

This experiment may shed some light on the impact of speech-recognition accuracy for SLS performance, if we compare SLS performance with the SRI and BBN recognizers, holding NL processing constant. The improvement of the SLS weighted error from 45.4% to 39.9% represents a error reduction by a factor of 0.12, and was obtained was obtained by running the NL system on input data for which the word error rate on class A and D sentences was improved from 8.4% to 6.2%, an error reduction factor of 0.26. The corresponding sentence error rates were 44.5% and 34.6%, for an error reduction factor of 0.22.

Although the NL processing in TRAVELOGUE is designed to be robust in the face of recognition errors, it is clear that the point of diminishing return on recognition accuracy has not yet been reached, and significant improvements can be obtained if these error rates can be reduced still further.

We did one other experiment with the combination of BYBLOS and TRAVELOGUE, in which we took the BYBLOS SPREC test output and ran it through TRAVELOGUE using the parameter settings that we now believe to be optimal as a result of the experiments reported in the preceding section. This was a combination

of the "always answer" context-handling strategy with the "relaxed cutoff" parameter settings. We felt that this would represent the best performance the system was currently capable of without increasing the basic underlying competence. In this experiment we obtained 495 correct answers, 77 wrong answers, and 117 without answers, for a weighted error of 39.16%.

7. ELIMINATING CLASS X SENTENCES

In addition to the above tests, we ran a test to evaluate the impact of a proposed change to the evaluation procedures to eliminate class-X sentences from the evaluation. Queries are classified as X for a variety of reasons, the most common being that the query lies outside the scope of the database. Although class-X utterances are not counted when computing the scores for NL and SLS evalutions, it may be the case that class-X queries that are clearly outside the scope of the system's processing capabilities could adversely impact the system's ability to track the context, and thus indirectly affect the system's test results.

If the inclusion of class-X sentences in the test were to make a large difference in the scores, it would call into question the success of the effort to eliminate the impact of processing class-X queries from the evaluation results.

To test the impact of class-X sentences on our system, we ran the system configured exactly as it was for the official test, except that all class-X sentences were excluded from consideration. We found that the weighted error decreased by 0.58 for the NL-only test and by 1.0 for the SLS test. While there is an observable "class-X effect," it seems to be relatively small with our system, and would only be noticeable with a processing strategy that based answering decisions on context uncertainty.

8. SUMMARY AND CONCLUSIONS

It is difficult to draw conclusions from these experiments about the efficacy of various parameter settings and processing strategies for improving performance on the evaluation. The results are in fact very similar, and could well be different with a different test set. It is possible to conclude with confidence only that the Maximum Precision strategy is unlikely to yield the lowest weighted error.

The results of these experiments were rather surprising in that we had originally believed that the parameter choices would have a more significant impact on the weighted error than what we observed. Indeed, the results show a surprising insensitivity to parameter choice. It seems to be the case that the weighted error metric disguises differences in system behavior. For example, the Maximum Precision and Maximum Recall strategies produce vastly different behavior on the SLS test: the Maximum Recall strategy answers almost 70 queries to which the Maximum Precision strategy gives no answer. Yet the difference in weighted error for the two strategies is less than one point.

For comparing performance across systems, it is desireable to have a metric for comparing performance across systems that is relatively insensitive to different answering strategies, and therefore has a better chance of truly reflecting the comprehensiveness of a system's coverage of the domain. These experiments demonstrate that the weighted error metric at least comes close to having that property — a fortunate consequence, because it was chosen primarily on the basis of its inuitive appeal. On the other hand, systems with specific characteristics are preferred for particular purposes. For example, when SRI uses its system for MADCOW data collection, it runs in a mode more closely approximating the Maximum Recall strategy, on the theory that producing some answer, even though not perfectly correct, will hold the user's interest and lead to a smoother flowing dialogue than would frequent "I don't understand" responses, even though the experiments indicate that such a strategy is suboptimal for evaluation. These experiments underscore the need to examine multiple properties of a system to arrive at conclusions regarding that system's overall effectiveness at solving user problems, as effectiveness can depend on factors other than the system's ability to obtain a low weighted error.

An important observation is that the five systems with the best scores in the NL evaluation differed by only 3.8 points. We have shown that our system can demonstrate a variation of more than 3 points in weighted error through the selection of different answering strategies holding the basic competence of the system constant. We would therefore be reluctant to conclude that the scores achieved on this benchmark test indicate a clear difference among these five systems in basic competence.

We found it interesting that the Always Answer context strategy would have produced the best results on this evaluation, because this is the most reasonable strategy to employ in a system intended to interact with a user, rather than merely scoring high on the evaluation. If the goal is to evaluate systems under conditions that approximate as much as possible their conditions of use in the real world, it is reassuring that behavior appropriate to the real world would not be inappropriate for the evaluation.

REFERENCES

1. Butzberger, J. et al., "Modeling Spontaneous Speech Effects in Large Vocabulary Speech Applications", Proceedings of the 1992 DARPA Speech and Natural Language Workshop.
2. Grosz, B. and Sidner, C., "Attentions, Intentions, and the Structure of Discourse," *Computational Linguistics*, Vol. 12, No. 3, 1966.
3. Jackson, E. et al., "A Template Matcher for Robust NL Interpretation," Proceedings of the 1991 DARPA Speech and Natural Language Workshop, pp. 190-194.
4. Murveit, H. et al., "Performance of SRI's Decipher Speech Recognition System on DARPA's ATIS Task," Proceedings of the 1992 DARPA Speech and Natural Language Workshop.
5. Murveit, H. et al., "Reduced Channel-Dependence for Speech Recognition," Proceedings of the 1992 DARPA Speech and Natural Language Workshop.