A Proposal For The Implementation Of

Real-time Speech Recognition Systems

Introduction

In order to build useful applications based on continuous speech recognition systems, such systems have to run in real time on common platforms. However, though most research has focused on various fast decoding algorithms with the tradeoff of small degradation, this thesis will investigate the overall system performance. Hence, the recognition performance including both the front end and the decoder will be addressed.

The recognizer presented here is derived from the existing ISIP speech recognition toolkit. All algorithms will be developed and tested on Resource Management (RM) database (1000 word, bigram perplexity of 60). Using the techniques presented in this thesis, the recognition system for RM task will run at 1 time real time (xRT), 6 times faster than the original system. It will be also shown that these techniques can be ported to other tasks, such as WSJ0 corpora (5k word, bigram perplexity of 147).

Real-Time Front End

One contribution of this thesis work is to achieve a non-latency front end. In speech recognition front end, Energy Normalization (EN) and Cepstral Mean Subtraction (CMS) algorithms are applied to improve the recognition performance. However, both techniques require the complete utterance to compute the maximum energy and cepstral mean. The computation of feature extraction can be very fast, but waiting for arrival of the whole utterance results in at least 1 xRT latency. Therefore the filter-based version of EN and CMS will be addressed to remove this drawback. Such normalization algorithms can be implemented to extract features on the fly. The experiments shown in Table 1 verify that such algorithms are comparable to the standard implementation of EN and CMS in speech recognition performance.

Exp	Front-End Condition	WER
01	Baseline: with both standard EN & CMS	3.7%
02	with standard EN & without CMS	3.8%
03	with standard CMS & without EN	4.1%
04	without EN and CMS	4.3%
05	Real-time EN (α) & standard CMS	3.5%
06	Real-time CMS (β) & standard EN	3.7%
07	both real-time EN (α) & CMS (β)	3.5%

Table 1: Experiments on RM database

For these experiments, the features are extracted with different front-end conditions and then the systems are retrained and tuned to optimum. Experiment 01 provides a baseline with 3.7% in WER. Experiments 02 - 03 show how EN and CMS impact the recognition performance: without *Energy Normalization*, the WER increases by 11% relative; without EN and CMS, the WER goes up by 16% relative. Experiments 05 - 07 show that after either or both standard EN and CMS algorithms are replaced by real-time EN and/or CMS, the systems can still achieve no degradation compared to the baseline. In the meantime, the adjustment of the regression factors α, β related to the respective algorithm will be also addressed in other experiments.

However, experiment 02 does not show CMS makes much difference, because CMS is primarily a channel normalization technique, but for a RM database the training and testing conditions are matched. The importance of CMS will be addressed with the discussion of the algorithm itself.

Real-time Decoder

Another contribution of this thesis work is to make a 6 xRT recognizer run on an 800 MHz processor, down to a near-1 xRT system with a certain threshold of degradation. Different approaches to reduce the runtime are explored. The results of some preliminary experiments are shown in Table 2. The first experiment (Experiment 01 in Table 2) is the **baseline** system, which is tuned to optimum (3.7%) on the development test set. To take this baseline system to near real-time, five methods are explored for increasing the decoding speed as following.

Exp	Condition	Lattice WER	WER	x Real-Time			Speedup (%)
				p1	p2	total	
01	baseline: 1-pass cross-word decoding	n/a	3.7%	6.1	n/a	6.1	n/a
02	2-pass decoding	0.6%	3.8%	3.14	0.1	3.24	47%
03	(02) + beam pruning	1.1%	4.1%	1.50	0.05	1.55	52%
04	(03) + NNA fast Gaussian	1.1%	4.1%	1.34	0.05	1.39	10%
05	(04) + phoneme lookahead	1.2%	4.1%	0.99	0.05	1.04	26%

Table 2: Preliminary experiments to yield a real-time system on RM

- **Two-pass decoding strategy:** word graphs are generated using word-internal models and then rescored using cross-word models. Thus, the large search explosion when crossing the word boundaries during cross-word decoding can be avoided. This strategy produces 47% speedup with only 1% WER loss (Experiment 02 in Table 2) compared to the baseline - one-pass cross-word decoding.

- **Search-Space Pruning:** search issues like the beam pruning, maximum number of active word ends, and maximum number of active phonemes, will be addressed. The main idea behind the algorithms is to decrease the recognition time by reducing the search space (Experiment 03 in Table 2). How to systematically tune these knobs will be interpreted in the thesis.

- **Fast Gaussian Evaluation:** Gaussian computation takes up a major portion (15 - 50%) of the overall recognition time. In this thesis, a *Near Neighbor Approximation* algorithm is developed to speed up the gaussian computation by 10% without degradation (Experiment 04 in Table 2). However, the Generalized-BBI algorithm [Srivastava, 2002] is a better approach to speed up Gaussian evaluation, approximately by 40% without degradation. In the future work, we will incorporate this work into our system.

- **Phoneme Lookaheads:** Another important technique in reducing the recognition effort is the use of phonetic lookaheads. Their job is to predict how well new phones will survive in the next couple of frames and then decide whether or not this transition is worth trying. This algorithm is very promising and yields 26% speedup with negligible degradation (Experiment 05 in Table 2).

- **Frame-Rate Reducing:** Reducing the frame rate is also a way to speed up the recognition. This technique requires the change of frame duration and window duration during feature extraction and model retraining. The method yielded a considerable speed-up in the MITRE project: it resulted in a 1 xRT system with 5.0% WER compared to the baseline with 3.4% WER at 8.3 xRT on a 600 MHz machine. However, this method should only be used if very high recognition speeds are required. Therefore, the final real-time system do not have this technique incorporated.

Since these approaches are sufficiently orthogonal to each other, the individual gains add up to the total required speed-up. Each section shows the possible speed-ups based on the baseline system. By combining these techniques, we have developed a real-time system with 4.1% WER on RM database. Compared to the baseline system, the real-time system only has 0.4% absolute degradation. These techniques will be also applied to another task - WSJ0 database.

The future work includes incorporating the BBI algorithm into the system and integrating the real-time front -end and real-time decoder together into the Communicator framework.

APPENDIX: Table of Contents

III. The Test Environment
- 3.1 The ISIP Recognition Toolkit
- 3.2 The Resource Management Task
- 3.3 The Wall Street Journal (WSJ0) Task
- 3.4 Baseline System

IV. Real-Time Front-End Normalization Techniques:
- 4.1 Filter-based Energy Normalization
- 4.2 Filter-based Cepstral Mean Subtraction
- 4.3 Real-time Front-End Summary

V. Real-Time Decoding Techniques:
- 5.1 Two-Pass Decoding
- 5.2 Pruning the Search Space
- 5.3 Near Neighbor Approximation Fast Gaussian Computation
- 5.4 Phoneme Lookaheads
- 5.5 Skipping Input Frames
- 5.6 Real-Time Decoding Summary

VI. Conclusion and Future Direction