A Multi-Pass, Dynamic-Vocabulary Approach to Real-Time, Large-Vocabulary Speech Recognition

I. Lee Hetherington

Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139 USA
ilh@csail.mit.edu

Abstract

We present a multi-pass approach to real-time, large-vocabulary speech recognition in which we dynamically manipulate the vocabulary between passes. For recognition tasks where subsets of the vocabulary can be triggered by the occurences of other words or phrases, a combination of unknown word modelling and vocabulary refinement can be utilized to attack large-vocabulary tasks with relatively small active vocabularies. We evaluate this approach within the JUPITER *weather information domain by enabling recognition of all 30,000 city-state pairs within the USA. By maximally precompiling the static and dynamic portions of our search space using finite-state transducers (FSTs), we splice dynamic-vocabulary components on-demand during decoding with negligible speed impact while enforcing cross-word context-dependent constraints. We find that a dynamic-vocabulary system can compete quite favorably with a single-pass, large-vocabulary system. For even larger vocabularies (e.g., street addresses), static compilation may be infeasible, making a dynamic-vocabulary approach necessary.*

1. INTRODUCTION

Traditionally, most speech recognition systems treated the vocabulary as fixed. However, a system with a flexible vocabulary may have advantages in terms of reacting to dialogue state, user customization, and novel words encountered within an application. In this paper we present a multi-pass approach to certain types of large-vocabulary recognition tasks by refining the vocabulary between passes, while operating in real time. For recognition tasks where subsets of the vocabulary can be triggered by the occurence of other classes of words or phrases, a combination of unknown word modelling and rapid vocabulary refinement can allow a large-vocabulary recognition task to be attacked with a much smaller active vocabulary.

In this paper we evaluate the approach within the JUPITER weather information domain [1] by enabling all 30,000 cities in the USA to be spoken as a city-state pair (e.g., "Will it snow in Ypsilanti, Michigan tomorrow?"). In contrast, the normal JUPITER system can recognize approximately 500 cities worldwide. We have previously performed some preliminary experiments using such an approach within a restaurant conversational system [2] that was developed and demonstrated Spring 2003, but the evaluations in the present paper utilize a much larger vocabulary, real users, and are constrained to real-time speed.

This research was sponsored in part by industrial consortia supporting the MIT Oxygen Alliance and the MIT Spoken Language Systems Group Affiliates Program.

Venkataraman et. al presented a very similar multi-pass technique within the context of spoken address recognition [3], although it does not appear that recognition speed was controlled when comparing the static and multi-pass system accuracies. Other work, for example that of Maskey et. al [4] demonstrate that constraining the vocabulary and language model based on other metadata (e.g., caller ID) can have powerful effects on recognition accuracy and they utilize an FST framework for splicing in subgrammars. In our work, we are not making use of any such external metadata, but rather seeking to constrain subgrammars of subsequent passed based on what is recognized in other parts of the same utterance in earlier passes.

2. ARCHITECTURE

2.1. Multi-Pass Approach

We propose to recognize some classes of large vocabularies by utilizing multiple passes and vocabulary refinement, a general idea presented to cope with many compound and inflected words in large-vocabulary recognition of broadcast news [5]. The basic idea is to make use of contextual information within the utterance itself to trigger the addition of new words or phrases to the recognizer. (Of course, previous utterances in the dialogue could also be used.) If the task vocabulary has hierarchical structure, this structure can be exploited by such a multi-pass approach.

Our approach in the first pass is to detect higher-level entities while skipping over any large-vocabulary, lower-level entities. For example, in an utterance containing city-state or street-city-state phrases, the first pass would seek to detect the state name while skipping over as yet out-of-vocabulary street or city names. We rely on an out-of-vocabulary (OOV) word model to handle any words or phrases that are at that time not yet active in the vocabulary, in essence as a type of "filler" model. After we detect one or more higher-level triggers in the N-best list, we consult a database to retrieve lower-level entities associated with the triggers and add them to the vocabulary. Thus, in our example, when the second pass starts, we will have activated all the cities associated with the detected state(s). This process can continue until the vocabulary hierarchy has been fully traversed.

In our decoder, we compute acoustic model scores on-demand and store them in a cache. Any model scores evaluated during the first pass do not need to be re-evaluated, thus significantly speeding up subsequent recognition passes. Furthermore, the use of an OOV model containing virtually all acoustic models serves to help fill the cache in the first pass.

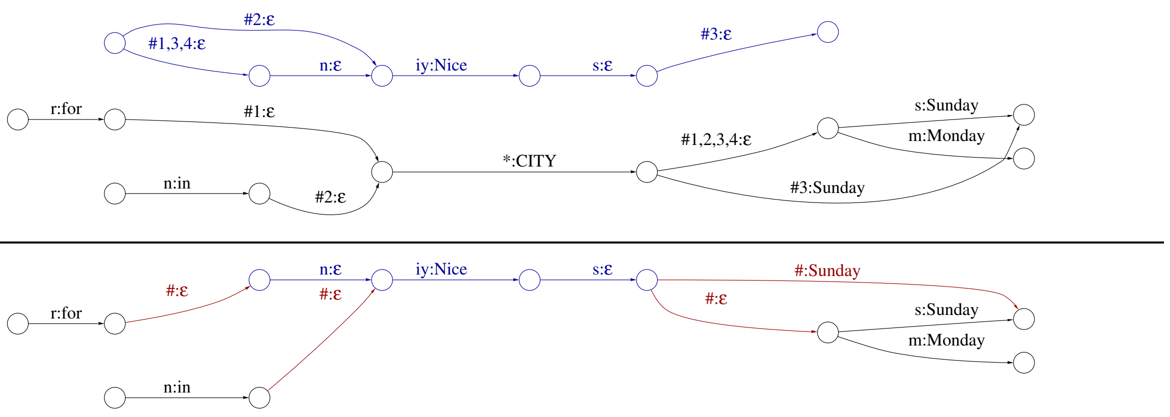

Figure 1: Example showing how auxiliary labels aid context-dependent FST splicing. The top shows a sample R_c containing a single word "Nice" and a part of R referring to a dynamic CITY class. The label "*:CITY" serves as a placeholder for the dynamic class contents. The bottom shows the result of the splice, where the auxiliary labels $\#_i$ have been appropriately matched, with resulting $\#$ labels indicating word boundaries. For example, for the phrase "in Nice Sunday," the /n n/ sequence is realized as [n] and the /s s/ sequence as [s] or [s s], demonstrating how cross-word context-dependent constraints can be enforced across a dynamic splice point. To simplify the figure we have applied only P to yield phonetic labels, as opposed to applying $C \circ P$ to yield context-dependent acoustic model labels.

2.2. FST Formulation

We make use of a finite-state transducer (FST) [6] formulation within our SUMMIT speech recognizer [7]. Using our own publicly available FST toolkit [8], we combine the recognition constraints into a single FST $R = C \circ P \circ L \circ G$, with G representing the grammar or language model, L the lexicon, P the phonological rules mapping phonemes to phones, and C the context-dependent model assignments.

2.3. Out-of-Vocabulary Modelling

Our approach depends critically on an unknown, or out-of-vocabulary (OOV) word model [9]. Within the lexicon L, the OOV model allows any combination of phonemes with a phonemic bigram constraint. The n-gram grammar G is trained with OOV examples, allowing G to provide constraints on likely locations of OOV occurrences during decoding.

2.4. Vocabulary Manipulation

We have the capability to rapidly reconfigure SUMMIT's vocabulary and grammar through the use of operations on the recognition FST R. The technique we have employed is to define "hooks" within R that allow for the on-demand inclusion of component FSTs that are modified based on dialogue or within-utterance context. It is the latter, *within-utterance context,* that was used to manipulate the vocabulary in the experiments of Section 3.

There are four primary issues related to manipulating the recognition FST R. First, since R plays a very active role in the recognition search, care must be taken to reduce any recognition speed degradation due to dynamic-vocabulary capabilities. Second, any splicing together of R and dynamic component FSTs must enforce all cross-word context-dependency constraints. In the SUMMIT recognizer, such cross-word constraints are contained with $C \circ P$, as both context-dependent model assignment C and context-dependent phonological rules P can cross word boundaries. Third, the language model (e.g., for an n-gram) must account for vocabulary or grammar changes in order to remain a valid probability model. Finally, if the potential vocabulary cannot be known in advance, a method to generate pronunciations for novel words must be used.

We address these issues within SUMMIT as follows. We achieve negligible decoding speed degradation by precompiling FST R as much as possible. Cross-word context-dependency contained within $C \circ P$ is applied fully where context is known, and auxiliary labels are utilized at dynamic splice points to account for all possible contexts that could arise. Thus, most of R can be statically expanded to the context-dependent acoustic model level. During decoding, dynamic class FSTs R_c are spliced into R, using the auxiliary labels to enforce constraints. Finally, we allow vocabulary and/or grammar manipulation only *within* n-gram classes. We compute n-gram probabilities as $P(w_i|c(w_i))P(c(w_i)|c(w_{i-1}), \ldots, c(w_{i-n+1}))$, where $c(w)$ represents the class of word w, and thus we need only update $P(w|c(w))$ when we update class c, which we can encode within the FST R_c representing the class. Finally, we make use of letter-to-sound capabilities described in [10] to generate phonemic pronunciations of words not in our dictionary.

Our approach to compilation of recognition transducer R and dynamic class components R_c and how they are dynamically spliced together on demand is similar to that of [11], but it is more general in the types of context dependency that can be handled. Our current approach is more general in that any context-dependency encoded with an FST can be utilized, and is outlined in Figure 1. We first compute FST $D = C \circ P$, containing *all* context-dependency. We then augment D with auxiliary labels of the form $\#_i$ at the boundaries of dynamic splice points to form D'. This D' is used to compute the *static* portion of $R = D' \circ L \circ G$, which has been expanded by D' to context-dependent acoustic models where possible, with auxiliary $\#_i$ labels at the edges of where dynamic classes will later be instantiated. When building R_c, the FST representing a dynamic class c to be spliced into R, we similarly compute $R_c = D' \circ L_c \circ G_c$. G_c in this case is the within-class n-gram probability plus any word sequence constraints within the class, and L_c is the within-class lexicon. During decoding, such R_c components are spliced into R, making sure that auxiliary labels across splice points are compatible, thereby enforcing valid cross-word context-dependent splices.

3. EXPERIMENTS

3.1. The JUPITER Domain

The data used for most of the training and all of the experimental evaluation came from telephone calls to the JUPITER weather information system [1]. JUPITER is a conversational system that can respond to natural language queries about weather forecasts for approximately 500 cities worldwide. Because of the mixed-initiative dialogue strategy employed, callers have great freedom in how they can express their requests, forcing JUPITER to handle highly spontaneous, continuous speech.

Our present goal in JUPITER is to allow the recognition of all 30,000 cities within the USA for which we can obtain weather forecast data from WSI Corp. (e.g., "Will it rain in St. Charles, Illinois tomorrow?"). In JUPITER we have approximately 70 n-gram word and phrase classes, including CITY, STATE, and CITY-STATE (e.g., "Boston," "Massachusetts," and "Boston, Massachusetts," respectively). In the first pass, we empty the CITY and CITY-STATE classes with the intent of forcing recognition of a city-state phrase as OOV followed by the state, thereby increasing the number of states detected within the N-best list. For any states detected, appropriate city-state entries are added to the CITY-STATE class. The CITY class is restored, and recognition is performed again with the refined vocabulary. In these experiments, we precompiled the CITY-STATE FSTs associated with each of the 50 states and stored them in compressed files on disk, with the intent that inactive parts of the vocabulary can be kept out of memory.

The acoustic models used in our experiments were trained on 140,000 utterances (120 hours) using a minimum classification error (MCE) criterion [12], yielding a total of 15,325 gaussian mixture components. The trigram class language model G was trained on 120,000 utterance transcripts, many containing out-of-vocabulary words modelled by an OOV model.

3.2. Baseline, Static, and Dynamic Systems

The *baseline* system has a fixed vocabulary of 2009 unique words. However, we make use of many compound words (e.g., "miami_florida" in the CITY-STATE class) to improve n-gram language modelling constraints, bringing the total lexical entries to 2912.

The *static* system is the baseline system with an additional 30,119 city-states added to the CITY-STATE class for a total of 30,562. The number of total unique words is 16,344, and the total number of lexical entries is 33,031. Within the CITY-STATE n-gram class, the within-class probabilities are uniform $(1/30562)$. Although these could be weighted (e.g., by population or based on the caller's area), we chose a uniform distribution for these experiments. As in the baseline system, the static system's vocabulary is fixed.

The *dynamic* system, multi-pass with vocabulary refinement, has the same potential vocabulary as the static system, but operates in the two-pass approach outlined above. Within the CITY-STATE class, the within-class language model probabilities for the city-states are uniform as in the static system, but the number of city-states active at any given time is much smaller, leading to higher within-class language model probabilities $1/N$.

	Test Set		
	A	B	C
baseline	17.2	8.8	23.9
static	17.1	9.1	7.0
dynamic	17.1	9.2	6.0

Table 1: % word error rate (WER) of three systems measured against three test sets. Test set B is all in-vocabulary for all three systems, and test set C is all in-vocabulary for the static and dynamic systems.

3.3. Evaluation

We used three test sets in our experiments, all of which were drawn from actual telephone conversations with our online JUPITER system. Test set A contains 1888 utterances, of which 430 contain artifacts such as noises and partial words, and 158 contain words outside of the baseline vocabulary. The OOV rate for the baseline vocabulary is 3.0% and for the expanded vocabulary is 1.9%. Test set B is a subset of A with 1313 utterances that contain no artifacts or out-of-vocabulary words. Test set C is separate and contains 327 utterances, each utterance containing at least one city-state, in which at least one word in the city name is outside of the baseline vocabulary, and all words are within the expanded vocabulary. One such utterance contains two city-states: "I would like to know what the weather is in Keizer, Oregon, and I would like to know what the weather is in St. Robert, Missouri."

In order to support fair accuracy comparisions, we have tuned beam-pruning parameters of the three systems so that they all run in real time on a 2.4GHz Pentium 4. Because the dynamic system has two passes, we had to use more aggressive pruning thresholds in both passes to achieve real-time performance. However, we hope that by automatically adapting the vocabulary to each utterance, such smaller beams do not hurt accuracy.

4. RESULTS

Table 1 displays the word error rate (WER) achieved for each of the three systems against each of the three test sets. For test set A, we see that all systems perform comparably, with the static and dynamic systems performing slightly better due to the larger vocabularies picking up a few city-states that were out-of-vocabulary for the baseline system. Test set B is completely in-vocabulary for the baseline system, and here we see only a small degradation in overall accuracy by enlarging the vocabulary of city-states. Test set C contains at least one out-of-vocabulary city word for the baseline system and is completely in-vocabulary for the static and dynamic systems. Test set C represents typical utterances we would like to handle with our expanded JUPITER vocabulary, and we find that in terms of WER the new static and dynamic systems perform quite well, even better than the baseline system does on test set B.

How well are we recognizing the new, larger set of city-states? Table 2 displays a city-state token error rate on set C, where a token is a complete city-state pair such as "Chagrin Falls, Ohio." A system must completely match the city-state for it to score correctly. By construction, the baseline system has a 100% token error rate on test set C, with most of the errors being deletions (substituting non- city-states in their place). We find that the multi-pass dynamic system outperforms the static system in this case, getting 16.5% of the city-states incorrect vs.

	%Err	%Sub	%Del
baseline	100.0	13.7	86.3
static	19.2	12.1	7.0
dynamic	16.5	15.2	1.2
dynamic-oracle-1	19.2	11.3	7.9
dynamic-oracle-2	14.9	14.3	0.6

Table 2: % error, substitutions, and deletions for only city-state tokens of test set C utterances. There were no insertions.

19.2% for the static system.

Analyzing the behavior of the dynamic system, we find that it detected 97.6% of the states in the first pass, proposing on average 1.4 states per utterance. This activated an average of 1115 city-states for the second pass. This is a 30-fold reduction in the average number of active city-states vs. the static system.

We hypothesize two likely reasons why the dynamic system outperformed the static system on set C: (1) the dynamic system has a smaller, less confusable search space, and (2) the uniform within-class language model score ($1/N$) is higher for the dynamic system because N is smaller due to fewer active city-states. To help shed light on the relative importance of these two effects we performed an experiment with the dynamic system in which we assume its first pass, detecting states names, is perfect. In Table 2, the dynamic-oracle-1 and -2 systems are given the correct state name(s) for the second pass, with the only difference being the within class language model probabilities; their search spaces are otherwise identical. The -1 system, which achieved a 19.2% city-state error rate used the static system's $1/30562$ probability for each city, and the -2 system, which achieved a 14.9% city-state error rate used the $1/N$ relevant to the given state(s). This appears to be clear evidence that the within-class language model score is largely responsible for the observed performance difference.

While we did aim for *overall* computation in real-time, it is very likely that a multi-pass approach will have a higher latency. We observed that approximately 50% of the computation used by the dynamic system occurs after the end of the utterance and directly contributes to latency. By comparison, for the static system, this fraction is generally below 10%. However, for these experiments, we did not tune our multipass system with the goal of minimizing latency.

5. Conclusion

We have presented an approach to real-time large-vocabulary recognition tasks utilizing multiple passes and rapid vocabulary refinement. In particular, it is applicable to tasks where detecting certain classes of words can lead to constraints in other classes. In this paper, we have explored this approach within the JUPITER weather information domain in which all cities within the USA are speakable if accompanied by the associated state names. By detecting state names in the first pass, relevant city names are activated, or added to the vocabulary, for the second and final pass.

We found that with all systems operating at 1x real-time total computation, that the multi-pass, dynamic-vocabulary approach was competitive with both the original baseline and large-vocabulary static systems, despite operating with an active vocabulary typically 30-fold smaller than the static system. On utterances containing previously out-of-vocabulary city-states, the dynamic-vocabulary system performed the best, recognizing 83.5% of the new city-states.

While the static and dynamic systems described here performed comparably, it may be impractical to operate with a static vocabulary for very-large-vocabulary tasks recognizing terms such as name-city-state or street-city-state. For example, the recognition of street addresses in the USA of the form "32 Vassar Street, Cambridge, Massachusetts" requires very large vocabularies indeed. According to the 2000 US Census,[1] there are approximately 6.4M valid street-city-states, containing 1.4M unique street names, comprised of nearly 300K unique words. Having all of those street-city-states active in a first pass may have serious speed, memory, and accuracy implications. A multi-pass approach locating the state name in the first pass, the city name in the second pass, and finally the street in the third pass will almost certainly be preferable, and the results of [3] appear to support this hypothesis. We too have built a preliminary demonstration version of such a system, but have not evaluated its performance formally. We do find that it can often operate with a vocabulary of a few thousands words despite having a virtual vocabulary size of nearly 300K words.

6. References

[1] V. Zue, S. Seneff, J. Glass, J. Polifroni, C. Pao, T. J. Hazen, and L. Hetherington, "Jupiter: A telephone-based conversational interface for weather information," *IEEE Trans. Speech and Audio Processing*, vol. 8, no. 1, pp. 85–96, Jan. 2000.

[2] G. Chung, S. Seneff, C. Wang, and L. Hetherington, "A dynamic vocabulary spoken dialogue interface," in *Proc. Intl. Conf. on Spoken Language Processing*, Jeju, Oct. 2004.

[3] A. Venkataraman, H. Franco, and G. Myers, "An architecture for rapid retrieval of structured information using speech with application to spoken address recognition," Dec. 2003.

[4] S. Maskey, M. Bacchiani, B. Roark, and R. Sproat, "Improved name recognition with meta-deta dependent name networks," in *Proc. Intl. Conf. on Acoustics, Speech, and Signal Processing*, Montreal, May 2004, pp. 789–792.

[5] P. Geutner, M. Finke, and P. Scheytt, "Adaptive vocabularies for transcribing multilingual broadcast news," in *Proc. Intl. Conf. on Acoustics, Speech, and Signal Processing*, Seattle, May 1997, pp. 925–928.

[6] F. Pereira and M. Riley, "Speech recognition by composition of weighted finite automata," in *Finite-State Language Processing*, E. Roche and Y. Schabes, Eds., pp. 431–453. MIT Press, Cambridge, 1997.

[7] J. Glass, "A probabilistic framework for segment-based speech recognition," *Computer Speech and Language*, vol. 17, no. 2-3, pp. 137–152, April-July 2003.

[8] L. Hetherington, "The MIT finite-state transducer toolkit for speech and language processing," in *Proc. Intl. Conf. on Spoken Language Processing*, Jeju, Oct. 2004.

[9] I. Bazzi and J. Glass, "Modeling out-of-vocabulary words for robust speech recognition," in *Proc. Intl. Conf. on Spoken Language Processing*, Beijing, Oct. 2000, pp. 1613–1616.

[10] G. Chung, C. Wang, S. Seneff, E. Filisko, and M. Tang, "Combining linguistic knowledge and acoustic information in automatic pronunciation lexicon generation," in *Proc. Intl. Conf. on Spoken Language Processing*, Jeju, Oct. 2004.

[11] J. Schalkwyk, L. Hetherington, and E. Story, "Speech recognition with dynamic grammars using finite-state transducers," in *Proc. Eurospeech*, Geneva, Sept. 2003, pp. 1969–1972.

[12] E. McDermott and T. J. Hazen, "Minimum classification error training of landmark models for real-time continuous speech recognition," in *Proc. Intl. Conf. on Acoustics, Speech, and Signal Processing*, Montreal, May 2004, pp. 937–940.

[1] http://www.census.gov/geo/www/tiger/tiger2k/tgr2000.html