A TAIL-SHARING WFST COMPOSITION ALGORITHM FOR LARGE VOCABULARY SPEECH RECOGNITION

Diamantino Caseiro, Isabel Trancoso

L^2F Spoken Language Systems Lab.
INESC-ID/IST
Rua Alves Redol 9, 1000-029 Lisbon, Portugal
{dcaseiro, Isabel.Trancoso}@l2f.inesc-id.pt

ABSTRACT

This paper presents an algorithm for approximating minimization in the context of the weighted finite-state transducers approach to large vocabulary speech recognition. The algorithm is designed for the integration of the lexicon with the language model and performs composition, determinization and pushing in one step. Furthermore, it uses tail-sharing in order to approximate minimization. Our results show that it is a good approximation to explicit minimization, with the added advantage that it can be used "on-the-fly" in a dynamic decoder.

1. INTRODUCTION

Weighted finite-state approaches have been gaining popularity in the speech recognition community [1, 2, 3, 4] due to advantages such as flexibility [5, 6, 4] and efficiency [3, 7, 2].

The approach can be briefly described in the following way: all knowledge sources, such as the lexicon or the language model (LM), are encoded as weighted finite-state transducers (*WFSTs*); the knowledge sources are combined using *WFST* composition, in a very large integrated static network, that is then optimized using well founded algorithms such as determinization, minimization and pushing. The flexibility of the approach comes from the uniformity of representation and combination of the knowledge sources, allowing the easy integration of novel sources. The efficiency comes from the optimization algorithms that make very good use of the sparsity of the integrated network.

The main problem with the approach has to do with scalability. This problem has two aspects: the first one is that the memory required to determinize and optimize the integrated network is much larger than the size of the resulting network; the second one is the large size of the static network in runtime.

In previous work [8] we proposed a *WFST* composition algorithm, specialized for the integration of the lexicon with the language model, to get around the first aspect of the problem. Later, in [9], we addressed the second aspect by employing an optimized version of the algorithm to perform the combination and optimization of the knowledge sources in runtime and "on-the-fly". With this algorithm, we showed that *WFST* approaches could be used in a dynamic decoder which might have advantages when the original knowledge sources are adapted.

Our "on-the-fly" algorithm is only exact for some local knowledge integration and optimization operations such as composition and determinization. Other operations, of a global nature, have to be approximated.

One of these global operations is the *weight pushing* operation, that consists of redistributing weights throughout the network in order to reduce pruning errors, and consequently, improving performance through the use of tighter beams. In our algorithm we approximate the intra-word effect of pushing with the use of "language model lookahead", and the inter-word effect by applying "pushing" to the language model, in a preprocessing step.

The other main global optimization operation is *minimization*; its main effect is the reduction of the size of the network, but it also has a positive influence on the performance of the decoder.

We differentiate two main aspects of minimization: on one hand it merges large portions of the network corresponding to full words; on the other it merges the suffixes of words. We approximate the first aspect by minimizing the language model in a preprocessing step. In [10] we proposed a suffix sharing technique to address the second aspect, but it was not efficient enough for "on-the-fly" use in a decoder. In this paper, we propose an efficient technique that is equivalent when n-gram language models are used.

In the context of search using a static network, it was proposed by [11] that, since all words linking to the same LM-history conditioned lexicon-tree are the same, then, after language model lookahead optimization, the linear pronunciation suffixes (*tails*) of such words could be shared, thus reducing redundancy. The proposed method resembles this *tail-sharing* technique in that it allows the sharing of pronunciation suffixes by instances of the same word.

In the next section we describe our composition algorithm and the structure of the search space it generates. In section 3 we present the new tail-sharing algorithm, followed by the results of its application to an European Portuguese broadcast news task. Finally, the main conclusions are presented.

2. COMPOSITION OF THE LEXICON WITH THE LANGUAGE MODEL

Our algorithm for the integration of the lexicon with the language model is based on a theorem [12] that states that the transducer resulting of the composition of deterministic transducers is also deterministic. Using this result, we avoid the explicit determinization of the composition of the lexicon with the language model by determinizing them independently. In practice, when the usual composition algorithm for *WFSTs* is used, this theorem is of little help, because of the generation of too many non-coaccessible[1]

[1] A non-coaccessible path is a "dead-end" path that does not reach a final state.

paths in the result, which make the method unpractical.

We avoid this problem by developing a specialized version of the *WFST* composition algorithm that only generates useful states. It relies on the use of a lexicon where every ϵ (or *empty*) output edge was previously tagged with the set of non-ϵ output labels directly accessible from that edge. Those sets are used to avoid following ϵ output edges of the lexicon when they do not lead to a useful path.

In traditional continuous speech large vocabulary recognition systems, the search space is sometimes described in terms of copies of the lexicon, while in *WFST* approaches an algebraic description is preferred. Our composition algorithm is equivalent to the use of explicit determinization for optimization of the search space, in the sense that the resulting networks represent the same relation and both are deterministic, but the precise distribution of weights and the location of output labels along paths may be different. This equivalence can help us gaining some insight into the determinized search space in terms of lexicon copies.

In the following discussion we assume that the lexicon *WFST* is organized in a loop, in which the initial state is both the initial and final state of all pronunciations. To exemplify, figures 1, 2, and 3 show a toy lexicon loop L^*, a language model G and their composition.

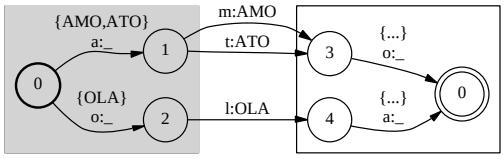

Fig. 1. Tagged lexicon loop L^*.

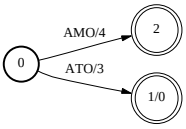

Fig. 2. Sample Language Model G.

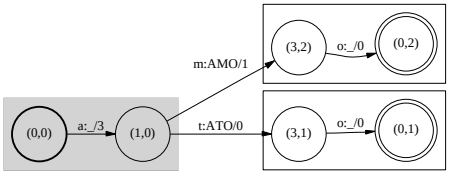

Fig. 3. Composition $L^* \circ G$.

In the figures, ϵ labels are represented as _, the set of all words is represented as $\{...\}$. For clarity, the initial state of the lexicon loop is shown twice in figure 1.

Each state of the composed *WFST* corresponds to a pair of states from the argument transducers (for example, state (1,0) corresponds to state 1 of the lexicon and to state 0 of the LM). We designate states with the form (i^l, q^g), where i^l is the initial state of the lexicon and q^g a language model state, as *anchor* states. Assuming that the lexicon includes pronunciations for all words in the language model, then anchor states (and indirectly, the language model) shape the global structure of the search space: the global structure consists of paths corresponding to the pronunciation of words connecting anchor states.

To help the local characterization of the search space, we designate states in the lexicon that are between the initial state and a non-ϵ output edge, as *prefix* states (shown inside a gray rectangle in figure 1). And we designate states in the lexicon between a non-ϵ output edge and the final state as *suffix* states (shown inside a white rectangle).

Analyzing the composed *WFST*, we see that two types of replication of the lexicon occur: one that starts in anchor states and replicates lexicon prefix states until non-ϵ output edges are found; and other that consists of suffix states, and progresses until the end of the pronunciation. In the first type of replication, edges (and states) of the lexicon are filtered by the input labels (words) of edges leaving the language model state, so that only edges of the lexicon which match a language model label either in its tagging set or in its output label are copied. In the second type, the filtering is done by the incoming words of the language model state that corresponds to the destination anchor state. In figure 3 three replication regions are marked, one prefix region, in gray, and two suffix ones, in white rectangles.

It is obvious that the topology of the lexicon *WFST* is crucial in determining the shape, size and performance of the search space. If the lexicon is deterministic and the identity of the word is produced in the last edge of the pronunciation, then no suffix sharing will take place among words arriving at the same language model state. If the lexicon is deterministic and minimal, as the one shown in figure 1, then some suffix sharing will occur, even among different words, if the language model *WFST* allows it.

Figure 4 shows an example language model G' that when composed with the lexicon loop generates the network shown in 5. As this example shows, the suffix sharing will not be, in general, complete. In order to more closely approximate minimization we would like to obtain the network shown in figure 6.

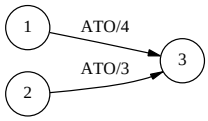

Fig. 4. Sample language model G'.

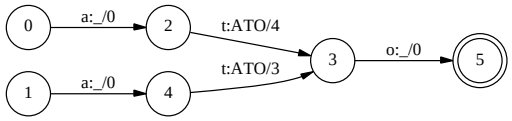

Fig. 5. Composition $L^* \circ G'$.

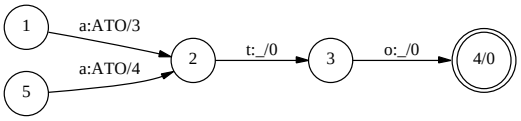

Fig. 6. Desired composition $L^* \circ G'$, with complete sharing of suffixes.

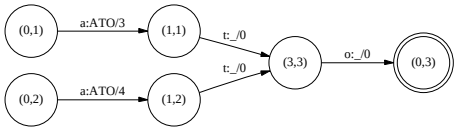

Fig. 7. Composition $L^* \circ G'$ with pushing.

2.1. Pushing of Weights and Output Labels

In the previous discussion, nothing was said about the distribution of weights in the network. If the composition algorithm, as described so far, is used, language model weights are incorporated into the composed network only in non-ϵ output edges. We improved our algorithm in order to spread language model weights throughout the replication of the prefix region of the lexicon.

When the algorithm verifies if a word from the language model is present in the set tagging an ϵ output edge of the lexicon, it keeps track of the edge with the best matching weight in the language model. The difference between this best weight and the language model weight that was spread from the previous anchor state is combined with the weight of the lexicon edge in order to produce the weight of the composed edge. To help this computation, the language model weight spread since the beginning of the pronunciation is kept in each state of the composed *WFST*.

When only one language model edge matches a lexicon edge, the output label is immediately produced. This has the effect of pushing the output labels toward the initial state. A boolean value is associated with each composed state indicating if the output label was produced or not, and is used to avoid its production more than once.

This improvement of the algorithm does not change the topology of the network, only the distribution of weights and output labels throughout its paths.

One effect of this redistribution of weights and output labels is that in the resulting network further sharing of suffixes will be easier. Figure 7 shows the composition of L^* with the language model G' using this improvement, where we see that now states (1,1) and (1,2) may be safely merged.

3. TAIL-SHARING *WFST* COMPOSITION ALGORITHM

It is difficult to know, in an "on-the-fly" algorithm that generates the search space state by state, such as ours, when two states like (1,1) and (1,2) can be merged.

In the algorithm described in [10], we proposed the use of two versions of the lexicon: one optimized for prefix sharing; and the other optimized for suffix sharing. The second version was used instead of the suffix region of the lexicon as soon as the output label was produced by pushing. The main problem with that approach was that, due to the need of finding correspondences between states of both lexicons, it was too slow for use in a dynamic decoder. Also, its ability of sharing suffixes of different words is useless when n-gram language models are used, as the edges entering an n-gram language model are either backoff edges labeled with ϵ or have all the same label. When n-gram language models are used, sharing the suffixes of the same word (tail-sharing) is enough.

The modification of the algorithm to allow tail-sharing consists of the use of a hash table containing composed states, indexed by a tuple $(lex\ state, lex\ output\ label, next\ anchor\ state)$. Whenever a state (l, g) is generated such that l is a prefix state, but the output label was already produced, then a lookup is made to the hash table to find a state on the same position in the lexicon, on a path of the same word and going to the same anchor state. If such a state is found, it is used instead of (l, g), if it is not found then (l, g) is entered on the hash table. The apparent ambiguity of indexing the hash table with both the word and the destination state is necessary to avoid incorrect behavior when using non n-gram language models that allow different words to enter the same state.

In figure 8 we see the effect of the tail-sharing composition algorithm. Assuming that state (1,1) was generated first and is thus present in the hash table, when state (0,2) is expanded and state (1,2) generated, the algorithm detects that (1,2) is equivalent to (1,1) and uses it as destination of the edge leaving (0,2). State (1,2) is ignored as it has no incoming or outgoing edges.

Our method differs from [11] in that it is restricted neither to a tree lexicon; neither to n-gram language models. Furthermore, it progresses forward state by state, while the original tail-sharing algorithm performs the sharing of tails by following the pronunciation backwards, from the leaves to the root of the pronunciation tree.

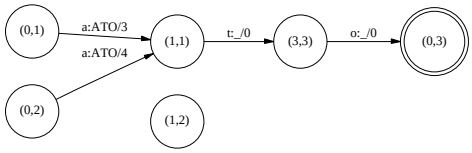

Fig. 8. Tail-sharing composition $L^* \circ G'$.

4. EXPERIMENTAL RESULTS

The recognition experiments described in this section were based on the European Portuguese broadcast news corpus collected in the scope of the ALERT European project [13]. The acoustic models used by our hybrid recognition system are based on the combination of various multilayer perceptrons[14]. The lexicon used contained 57k words, and was first converted to a deterministic *WFST* with 157,255 states and 221,321 edges and this was minimized resulting in a *WFST* with 58,837 states and 121,657 edges. The language model was obtained by interpolation of training set transcriptions with online newspaper texts. It is a 3-gram language model containing 3.7M n-grams. The language model was first converted to a pseudo-deterministic *WFST* with 1,961,814 states and 5,719,694 edges and then minimized to 528,879 states and 4,280,938 edges.

In the first experiment, we compared the search networks generated by the previous version of the composition algorithm, versus the one generated by the new tail-sharing algorithm, and the use of explicit offline transducer minimization. We used the minimized lexicon and language model *WFSTs*. Table 1 shows the dimension of the search spaces. We observe that the network obtained with tail-sharing algorithm has only 5% more states and 3% more edges than the minimum, and we also observe a huge reduction on the size of the network relative to the previous version.

In table 2 we show the performance resulting of the use of the networks at various beams. One thing we notice is that all networks produce the same word error rate (WER) at the same beam. This was expected for the previous and the tail-sharing algorithm, as the distribution of weights along paths is the same. The fact that

	Previous	*Tail-Sharing*	*Minimum*
States	16,612,807	4,596,489	4,383,798
Edges	21,270,815	8,925,409	8,698,667

Table 1. Dimension of search spaces.

	Previous		*Tail-Sharing*		*Minimum*	
Beam	xRT	WER	xRT	WER	xRT	WER
4.0	0.19	42.5	0.17	42.5	0.17	42.5
4.5	0.43	39.0	0.40	39.0	0.40	39.0
4.7	0.59	38.1	0.54	38.1	0.54	38.1
5.0	0.91	37.5	0.84	37.5	0.83	37.5
5.2	1.19	37.1	1.10	37.1	1.09	37.1
5.5	1.71	37.1	1.60	37.1	1.57	37.1
5.7	2.15	36.8	2.01	36.8	1.98	36.8

Table 2. Performance of the various search spaces.

is was also the same for the minimized network is an indication that the tail-sharing algorithm is a good approximation. Looking at the real time factor (xRT), we see that while the previous algorithm was 8 to 11% slower than the minimized network, the new one is only 0.5 to 1.5% slower. The behavior of the new algorithm is almost indistinguishable from the minimized network.

The main advantage of the tail-sharing algorithm versus explicit minimization is that it can be used "on-the-fly" in a dynamic decoder. We compared the tail-sharing algorithm with the previous one in a dynamic decoder. The WER values obtained were the same as before, and the real-time factors are shown in table 3. The observed improvement is around 4 to 7%.

Beam	*Previous*	*Tail-Sharing*
4.0	0.24	0.23
4.5	0.50	0.48
4.7	0.67	0.63
5.0	1.01	0.96
5.2	1.30	1.23
5.5	1.85	1.74
5.7	2.30	2.15

Table 3. "On-the-fly" performance.

5. CONCLUDING REMARKS

We have presented a tail-sharing approximation to minimization specially tailored for use in LVR. It provides an exact simultaneous composition and determinization of the lexicon and language model *WFSTs*. Furthermore, it also approximates pushing and minimization of the search space. The tail-sharing approximation permits the "on-the-fly" generation of search spaces very close in size and performance to the ones optimized by offline minimization. The algorithm is efficient enough for use in a dynamic decoder, with advantages of added scalability and adaptability relative to the use of static search networks.

6. ACKNOWLEDGEMENTS

The present work is part of Diamantino Caseiro's PhD thesis, initially sponsored by a FCT scholarship (PRAXIS XXI/BD/15836/98). This work was also partially funded by IST-HLT European program project ALERT, and by FCT national project IPSOM (POSI-34252/99). INESC-ID Lisboa had support from the POSI program of the "Quadro Comunitario de Apoio III".

7. REFERENCES

[1] M. Mohri, F. Pereira, and M. Riley, "Weighted Finite-State Transducers in Speech Recognition," in *ASR 2000 Workshop*, September 2000.

[2] J. Glass, T. Hazen, and I. Hetherington, "Real-Time Telephone-Based Speech Recognition in the Jupiter Domain," in *Proc. ICASSP '99*, Phoenix, Arizona, U.S.A., May 1999.

[3] H. Dolfing, "A Comparison of Prefix Tree and Finite-State Transducer Search Space Modellings for Large-Vocabulary Speech Recognition," in *Proc. ICSLP '2002*, Denver, Colorado, U.S.A., September 2002.

[4] M. Szarvas and S. Furui, "Finite-State Transducer Based Hungarian LVCSR with Explicit Modelling of Phonological Changes," in *Proc. ICSLP '2002*, Denver, Colorado, U.S.A., September 2002.

[5] T. Hazen, I. Hetherington, and A. Park, "FST-Based Recognition Techniques for Multi-Lingual and Multi-Domain Spontaneous Speech," in *Proc. Eurospeech '2001*, Aalborg, Denmark, September 2001.

[6] I. Bazzi and J. Glass, "Learning Units for Domain-Independent Out-of-Vocabulary Word Modelling," in *Proc. Eurospeech '2001*, Aalborg, Denmark, September 2001.

[7] S. Kanthak, H. Ney, M. Riley, and M. Mohri, "A Comparison of Two LVR Search Optimization Techniques," in *Proc. ICSLP '2002*, Denver, Colorado, U.S.A., September 2002.

[8] D. Caseiro and I. Trancoso, "On Integrating the Lexicon with the Language Model," in *Proc. Eurospeech '2001*, Aalborg, Denmark, September 2001.

[9] D. Caseiro and I. Trancoso, "Using Dynamic WFST Composition for Recognizing Broadcast News," in *Proc. ICSLP '2002*, Denver, Colorado, U.S.A., September 2002.

[10] D. Caseiro and I. Trancoso, "Transducer Composition for "On-the-Fly" Lexicon and Language Model Integration," in *Proc. ASRU '2001 Workshop*, Madonna di Campiglio, Trento, Italy, December 2001.

[11] F. Brugnara and M. Cettolo, "Improvements in Tree-Based Language Model Representation," in *Proc. Eurospeech '95*, Madrid, Spain, September 1995, pp. 2133–2136.

[12] M. Mohri, "Finite-State Transducers in Language and Speech Processing," *Computational Linguistics*, vol. 23, no. 2, pp. 269–311, June 1997.

[13] H. Meinedo, N. Souto, and J. Neto, "Speech Recognition of Broadcast News for the European Portuguese Language," in *Proc. ASRU '2001 Workshop*, Madonna di Campiglio, Trento, Italy, December 2001.

[14] H. Meinedo and J. Neto, "Combination of Acoustic Models in Continuous Speech Recognition Hybrid Systems," in *Proc. ICSLP '2000*, Beijing, China, October 2000.