SILENCE IS GOLDEN: MODELING NON-SPEECH EVENTS IN WFST-BASED DYNAMIC NETWORK DECODERS

David Rybach, Ralf Schlüter, Hermann Ney

Human Language Technology and Pattern Recognition, Computer Science Department,
RWTH Aachen University, 52056 Aachen, Germany
{rybach,schlueter,ney}@cs.rwth-aachen.de

ABSTRACT

Models for silence are a fundamental part of continuous speech recognition systems. Depending on application requirements, audio data segmentation, and availability of detailed training data annotations, it may be necessary or beneficial to differentiate between other non-speech events, for example breath and background noise. The integration of multiple non-speech models in a WFST-based dynamic network decoder is not straightforward, because these models do not perfectly fit in the transducer framework. This paper describes several options for the transducer construction with multiple non-speech models, shows their considerable different characteristics in memory and runtime efficiency, and analyzes the impact on the recognition performance.

Index Terms— LVCSR, WFST

1. INTRODUCTION

Acoustic models for non-speech events like silence and noise are a fundamental part of a speech recognition system. If the silence model does not match with non-speech parts of the signal, the system will produce insertion errors, while a vague silence model may cause deletion errors. Depending on the kind of audio data to be processed and the upstream audio segmentation, different kinds of non-speech events have to be considered, for example breath, laughter, hesitations, or background noise. For some systems it may therefore be beneficial to train separate models for these non-speech events. Such models require of course respective precise annotations in the training data.

The WFST framework offers a clear and consistent way of modeling the parts of a speech recognition system [1]. However, non-speech events do not perfectly fit in this framework, because they are usually not covered by the LM [2]. The search graph construction is implemented by a chain of token sequence expansions, from words down to HMM states. Hence, non-speech models need to be present also at the word level. If the non-speech tokens do not occur as labels in the LM transducer, they cannot appear in the decoder output (unless the decoder implements some less generic special treatment), which is necessary for some applications. Furthermore, the non-speech labeled arcs in the LM transducer require a weight, which is generally not consistent with the rest of the LM [3].

In this paper, we analyze different options for integrating multiple non-speech models in the individual transducers involved without modifying the generic decoder itself. Some of these options allow for a significant reduction in size of the LM transducer. We use a dynamic network decoder, which integrates the LM on-the-fly as needed during recognition, allowing us to deal with huge vocabularies and complex language models (LM) in a memory efficient way [4]. In contrast to fully expanded static search graphs, the LM transducer is kept separately, thus the reduction in transducer size results in lower memory consumption of the speech recognizer. We also describe specifics of the context dependency transducer construction, which improve the runtime efficiency.

The remainder of this paper is organized as follows. Section 2 briefly describes the decoder used. In Section 3 we detail the construction of the individual speech recognition transducers. Section 4 presents the experimental results, followed by conclusions in Section 5.

2. DECODER

In the WFST framework, the LM is represented by a transducer G, L is a phone to word transducer derived from the pronunciation dictionary, and C encodes the context dependency of the acoustic models. These transducers are combined by the finite-state operation of composition as $C \circ L \circ G$. The composed transducer has tied HMM labels on the input side and words as output labels. The HMM states are generated dynamically during decoding in our system. In the dynamic network decoder, the composition of $(C \circ L)$ with G is computed on demand (lazy evaluation) using special composition filters which provide on-the-fly pushing of labels and weights [5]. We use determinized and minimized L and C transducers and perform no further transducer optimizations. Our decoder is based on the OpenFst toolkit [6].

3. TRANSDUCER CONSTRUCTION

The non-speech models need to be considered in all three transducers. In this section, we describe the construction options for the G and L transducer as well as special treatments of context independent models in the C transducer.

3.1. Language Model

In a G transducer representing a commonly used n-gram LM, the states encode word histories h and the arcs are labeled with words w. The weight of an arc is the LM probability $p(w|h)$. The backing off structure is implemented by epsilon arcs to a state with reduced history size [7].

Non-speech events are not part of the LM because a) they do not occur in the text data used to estimate the LM, b) their occurrence generally does not depend on the predecessor words, and c) they are not useful for predicting following words. Nevertheless, as mentioned in the introduction, it may be necessary to integrate tokens for non-speech events in the G transducer. As silence and noise can occur at any position in the spoken word sequence, non-speech

This work has been funded in part by the Google Research Awards Program and was partly realized as part of the Quaero Programme, funded by OSEO, French State agency for innovation.

labeled arcs have to be reachable in G before and after any other arc. The non-speech arcs are constructed as self-loops, because they shall not modify the word history.

Adding loop arcs for all non-speech events to all states in the G transducer allows for the insertion of these tokens without any constraints and without reducing the context size. This construction heavily increases the transducer size though. An alternative producing significantly less arcs is to add non-speech loops only at the initial and the unigram (empty history) state. Thereby, non-speech events remove the history of subsequent words.

3.2. Lexicon

The L transducer has words as output labels and phones as input labels. If the non-speech tokens are included in the G transducer, L integrates the pronunciations for these tokens like any other word.

We can add the non-speech tokens as optional arcs after each word in L. These arcs have epsilon output labels, thus the non-speech tokens do not need to be handled by the G transducer. This construction introduces a limitation of recognizable sequences: after each word at most one non-speech token can be inserted, but not sequences of different non-speech tokens. The length of the non-speech events is not limited though, as loop transitions are added at the HMM state level. A simple solution for this problem would be to add self loops at the word end state. However, the computation of the reachable (output) label lookahead which is used for the composition filter (cf. [5]), requires that all cycles in the transducer contain at least one output label (implementation in OpenFst).

A compromise between constraining the sequences of non-speech events by adding optional arcs in L and reducing the LM context by introducing loops arcs at the unigram state in G, is to do both. Thereby, we can insert one non-speech token after each regular word without modifying the LM history and we can recognize longer sequences of several non-speech tokens by forcing a path trough the unigram state in G. This construction is reasonable, because it can be assumed that words following a longer pause do not depend on the preceding words. By adding only arcs for silence in L, we get a model which is very similar to the short silence model described in [2].

3.3. C Transducer

The C transducer is used to transform sequences of context independent (CI) phones, the input labels of the L transducer, to sequences of context dependent (CD) phone models by applying transducer composition as $C \circ L$. Therefore, C has CI phones as output labels and the CD units are used as input labels. The states in C encode the CI phone history (2 phones for triphone models). In the construction of an (output) deterministic transducer, as used in our system, the output label is the right context of the triphone model used as input label. This introduces a delay of CD labels by one symbol [8].

The models for non-speech events are usually context independent. In the C transducer, arcs with a CI pseudo phone (output) label s lead to a state $(*, s)$ with encodes just s as history. This state will have arcs for all phones π with the non-speech model ${}_{\#}s_{\#}$ (empty left and right context) as input label and state $(\#, \pi)$ as target. The state $(\#, \pi)$ represents an empty left context. See Figure 1 (a) for an illustration.

The composed transducer $C \circ L$ has at word ends a separate state for each phone model with empty right context and for each non-speech model. The non-deterministic outgoing arcs of this state all have the same input label (the respective non-speech model), as illustrated in Figure 1 (d). In contrast to arcs for CD models, whose input labels depend on the output label, the breakdown of output label dependent arcs is not required for CI models.

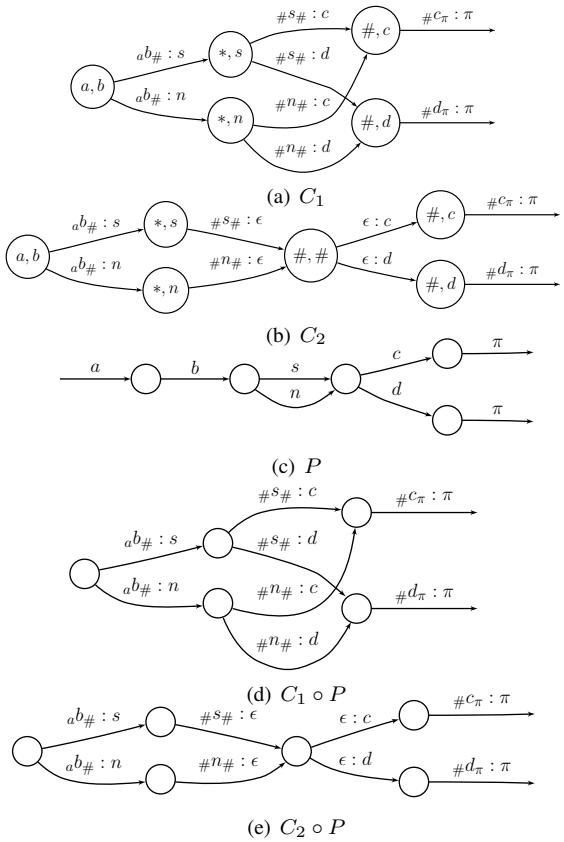

Fig. 1. C transducers with shifted CI phones (a), un-shifted CI phones (b); a phone sequence P (c) and the composition of the C transducers with P (d, e). s and n are CI non-speech pseudo phones, a, b, c, d, π are CD phones. Triphone models are denoted as ${}_{a}b_{c}$ with left context a, right context c.

The non-determinism can be eliminated by locally un-shifting or synchronizing the CI models in the C transducer. As shown in Figure 1 (b), we redirect the arcs with CI model input labels to the initial state (empty history) and replace the output label by epsilon. When composed with L, all arcs with non-speech input labels, whose predecessor states are deterministic now, will merge at one state (see Figure 1 (e)). A similar result can be obtained by applying transducer determinization to $C \circ L$. The determinization however would require the insertion of disambiguation symbols, because of the ambiguous transduction from HMM label strings to word label strings.

The described partially un-shifted construction does not reduce the size of $C \circ L$ much, but it improves the runtime performance in case of a acoustic model containing several CI models. During decoding, all state hypotheses for non-speech events at word boundaries will be recombined before being expanded to word initial states. Thereby, the number of active state hypotheses is significantly reduced and consequently the runtime performance.

4. EXPERIMENTAL RESULTS

We analyze the impact of the various options for the transducer construction on the recognition performance in terms of both recognition quality and efficiency.

4.1. Recognition System

We performed experiments on two different tasks: European Parliament Plenary Sessions (EPPS) in English and data from the Quaero Project in English, which contains broadcast news and unconstrained broadcast conversations in English. The systems differ in vocabulary size, complexity of the LM, the annotation of non-speech events in the training data, and the audio segmentation. The EPPS test data is segmented automatically resulting in a considerable amount of non-speech, while the Quaero data has a precise (manual) segmentation.

EPPS English: We use the system as described in [9]. The dictionary contains 53K words with 59K pronunciations, modeled using 45 phones and (unless noted otherwise) 5 non-speech pseudo phones (silence, hesitation, breath, laughter, general noise). The acoustic models (AM) consist of 4500 Gaussian mixtures modeling generalized triphone states with across word context dependency and using word boundary information. Triphones are modeled by 3-state HMMs, except for silence which has only one HMM state. The AM used for the first speaker independent recognition pass consists of 900K densities. A second AM, consisting of 800k densities, estimated using speaker adaptive training and discriminative training (minimum phone error criterion) was used in the second recognition pass, which applies speaker adaption using fMLLR. We used two 4-gram LMs of different size. The smaller LM contains 7.4M n-grams, the larger one has 25.8M n-grams. The test set comprises 644 segments with a total duration of 2.85h with about 27K words in total.

Quaero English: The Quaero ASR system uses 150K words with 180K pronunciations, using the same phoneme set as the EPPS system. The speaker independent AM consists of 1M densities for 4500 Gaussian mixture models. The MFCC features are augmented with phone posterior features. The 4-gram LM contains 50.4M n-grams. We used a simple one-pass decoding strategy for this task. A detailed description of the system is given in [10]. The test set used consists of 1482 segments with a total duration of 3.3h and contains about 40K words.

4.2. L and G Transducer

Table 1 illustrates the reduction in transducer size, in particular in the number of arcs in G. The table shows the size of $C \circ L$ built with and without optional non-speech arcs at word ends, as described in Section 3.2, plus the size of G with and without non-speech loop arcs as described in Section 3.1. The quantity of additional loop arcs added (number of non-speech tokens times number of states in G) is clearly relatively large, especially for complex language models like the larger EPPS LM and the Quaero LM. With an arc size of 16 bytes in memory, the additional arcs allocate around 520MB for the larger EPPS LM and 630MB for the Quaero LM. Considering that the LM often consumes the biggest fraction of memory of a speech recognition system, the transducer size reduction yields a noticeable decrease in memory requirements in practice. The increase in transducer size for additional optional non-speech arcs in $C \circ L$ is comparatively small.

Trading memory for speed or accuracy is easy in many cases. Interesting is therefore how the removal of non-speech arcs in G affects the recognition performance. The results achieved using the various construction options of G and L are shown in Table 2 for EPPS and Table 3 for the Quaero task. The baseline in these tables is a system which has loop arcs at all states in G. If loops are added only to the initial and the unigram states of G, the WER increases significantly. Adding non-speech arcs to L only yields an even worse result, because it is impossible to recognize sequences of different non-speech events this way. The increased number of insertion errors illustrates that. The combination of both options yields results nearest to the baseline, at least for the EPPS system. Adding only optional silence arcs to word ends in L deteriorates the results slightly.

Table 1. Transducer size for both systems. The number of arcs for the G transducer is given for the transducers with and without non-speech (non-sp.) loop arcs, for $C \circ L$ with and without optional non-speech arcs.

			arcs	
system	transducer	states	w/ non-sp.	w/o non-sp.
EPPS	$C \circ L$	$65.2K$	$253.4K$	$214.2K$
	small G	$1.8M$	$18.1M$	$9.2M$
	large G	$6.2M$	$62.7M$	$31.9M$
Quaero	$C \circ L$	$164.5K$	$515.5K$	$459.6K$
	G	$8.3M$	$100.1M$	$58.7M$

Table 2. Recognition results for the EPPS task using the small LM (upper part) and the large LM (lower part). Optional non-speech arcs were added in G at all states or only at the initial (i) and the unigram (u) state. Optional non-speech arcs in L were added either for all non-speech events, for silence (sil.) only, or not at all.

	non-sp. arcs		pass 1	pass2			
LM	L	G	WER	WER	sub.	del.	ins.
small	-	all	14.4	12.0	8.2	2.0	1.8
	-	i, u	16.7	14.1	9.7	2.6	1.7
	all	-	17.3	14.6	9.0	2.0	3.6
	all	i, u	14.5	12.1	8.3	2.0	1.8
	sil.	i, u	14.6	12.4	8.5	2.0	1.9
large	-	all	13.8	11.3	7.7	1.9	1.7
	all	i, u	14.0	11.4	7.9	1.8	1.7
	sil.	i, u	14.1	11.7	8.0	1.9	1.8

The Quaero system has slightly different characteristics. The duration of non-speech events in the test data is shorter here, because of the more precise segmentation. In addition, the non-speech models are less accurate due to less precise training data annotations. Even though the difference is quite small, the best option here is to add only silence instead of all non-speech tokens as optional word end arcs to L.

4.3. Non-speech Modeling

Instead of dealing with multiple non-speech events, we can also train just one non-speech model, whose mixture model accounts for the different acoustic realizations. We evaluated this option by comparing systems having one 3-state HMM model for noise in addition to the 1-state silence model with systems considering all non-speech tokens as described in Section 4.1. All systems were bootstrapped from the multiple non-speech system used for the experiments in the previous section, which might distort the results a little bit.

The baseline EPPS system has in total 12 non-speech (tied) state models (including one silence model), while the Quaero system has only 8. The lower number of non-speech models in the Quaero AM is mainly caused by a higher ambiguity in training data annotations and less observations for some of the events. The re-trained AM with one noise model has 4 non-speech HMM state models.

The results in Table 4 show that the EPPS system benefits from differentiated non-speech models, only slightly though. The Quaero system is not affected from pooling the noise models, which is not surprising because of the small difference in models as described above. All systems have noise and silence loop arcs on all G states.

Table 3. Recognition results for the Quaero task. Optional non-speech arcs were added in G at all states or only at the initial (i) and the unigram (u) state. Optional non-speech arcs in L were added either for all non-speech events, for silence (sil.) only, or not at all.

non-sp. arcs					
L	G	WER	sub.	del.	ins.
-	all	22.0	14.8	4.5	2.8
all	-	24.8	15.5	4.1	5.2
all	i, u	22.3	14.9	4.4	3.0
sil.	i, u	22.0	14.5	4.7	2.9

Table 4. Recognition results comparing acoustic models with one noise model vs. 4 noise models, both containing an additional one state silence model.

system	# noise m.	WER	sub.	del.	ins.
EPPS	4	14.4	9.8	2.7	1.9
	1	14.8	10.0	2.6	2.1
Quaero	4	22.0	14.7	4.5	2.8
	1	21.9	14.7	4.4	2.8

4.4. C Transducer

Figure 2 shows the effect of the C transducer construction described in Section 3.3 on the size of the active search space. The plot depicts the number of active state hypotheses (after pruning) in relation to the absolute number of word errors. Due to the earlier recombination of partial hypotheses after non-speech between words, the number of active state hypotheses is reduced by up to 40%.

The improvement in runtime efficiency is shown in Figure 3 (measured on a 2.8 GHz Intel Core2). Due to caching of acoustic scores, the decrease in real-time factor (processing time divided by audio duration) is lower than the reduction in search space size. Nevertheless, the RTF can be improved by up to 20%, which is noticeable in practice.

The construction also improves the runtime performance of the system with two noise models. The reduction of the active state space is smaller, but still around 20%.

5. CONCLUSION

Whether or not multiple non-speech models improve a speech recognition system depends on the targeted application, the training data, and the pre-processing. In our experiments the gain in recognition quality from including more than one non-speech model (in addition to a silence model) is small, if any.

If multiple non-speech models are present in the AM however, the construction using loop arcs at the unigram state in the G transducer combined with optional non-speech arcs at word ends in L, reduces the memory requirement significantly with a minor degradation in recognition accuracy. The adjusted construction of the C transducer decreases the size of the active search space and therefore improves the runtime efficiency of the decoder.

6. REFERENCES

[1] M. Mohri, F. Pereira, and M. Riley, "Speech recognition with weighted finite-state transducers," in *Handbook of Speech Processing*, J. Benesty, M. Sondhi, and Y. Huang, Eds. Springer, 2008, ch. 28, pp. 559–582.

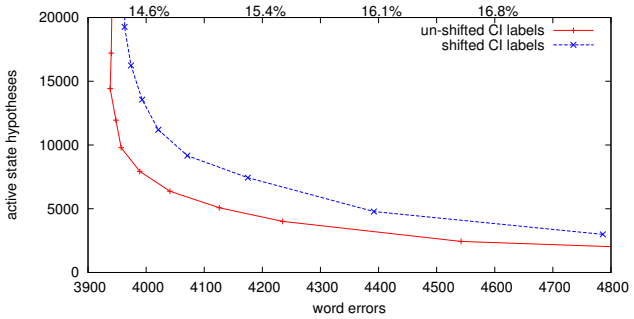

Fig. 2. Number of active states hypotheses as function of the absolute number of word errors for shifted and un-shifted CI labels in C. The corresponding WER is shown at the top of the plot.

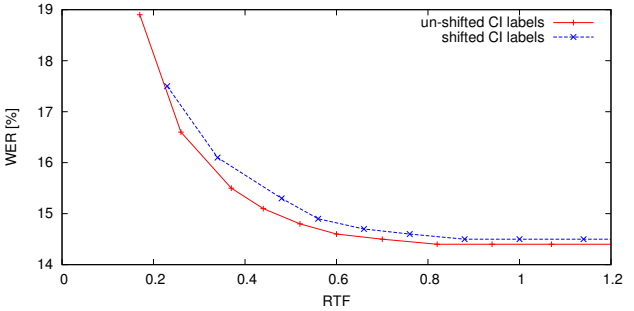

Fig. 3. WER vs. real-time factor (RTF) for search networks created with either shifted or un-shifted CI labels in C. The G transducer has loop arcs for all non-speech models.

[2] P. Garner, "Silence models in weighted finite-state transducers," in *INTERSPEECH*, Brisbane, Australia, Sep. 2008, pp. 1817–1820.

[3] C. Allauzen, M. Mohr, B. Roark, and M. Riley, "A generalized construction of integrated speech recognition transducers," in *INTERSPEECH*, Montreal, Canada, May 2004, pp. 761–764.

[4] D. Rybach, R. Schüter, and H. Ney, "A comparative analysis of dynamic network decoding," in *ICASSP*, Prague, Czech Republic, May 2011, pp. 5184–5187.

[5] C. Allauzen, M. Riley, and J. Schalkwyk, "Filters for efficient composition of weighted finite-state transducers," in *CIAA*, Winnipeg, Canada, Aug. 2010.

[6] C. Allauzen, M. Riley, J. Schalkwyk, W. Skut, and M. Mohri, "OpenFst: a general and efficient weighted finite-state transducer library," in *CIAA*, Prague, Czech Republic, Jul. 2007, pp. 11–23.

[7] C. Allauzen, M. Mohri, and B. Roark, "Generalized algorithms for constructing statistical language models," in *ACL*, Sapporo, Japan, Jul. 2003, pp. 40–47.

[8] M. Mohri and M. Riley, "Network optimizations for large-vocabulary speech recognition," *Speech Communication*, vol. 28, no. 1, pp. 1–12, May 1999.

[9] J. Lööf *et al.*, "The RWTH 2007 TC-STAR evaluation system for European English and Spanish," in *INTERSPEECH*, Antwerp, Belgium, Aug. 2007, pp. 2145–2148.

[10] M. Sundermeyer, M. Nussbaum-Thom, S. Wiesler *et al.*, "The RWTH 2010 Quaero ASR evaluation system for English, French, and German," in *ICASSP*, Prague, Czech Republic, May 2011, pp. 2212–2215.