Tied-State Mixture Language Model for WFST-based Speech Recognition

Hitoshi Yamamoto[1], *Paul R. Dixon*[1], *Shigeki Matsuda*[1], *Chiori Hori*[1], *Hideki Kashioka*[1]

[1]Spoken Language Communication Laboratory,
National Institute of Information and Communication Technology (NICT), Kyoto, Japan
hitoshi.yamamoto@nict.go.jp

Abstract

This paper describes a language model combination method for automatic speech recognition (ASR) systems based on Weighted Finite-State Transducers (WFSTs). The performance of ASR in real applications often degrades when an input utterance is out of the domain of the prepared language models. To cover a wide range of domains, it is possible to utilize a combination of multiple language models. To do this, we propose a language model combination method with a two-step approach; it first uses a union operation to incorporate all components into a single transducer and then merges states of the transducer to mix n-grams included in multiple models and to retain unique n-grams in each model simultaneously. The method has been evaluated on speech recognition experiments on travel conversation tasks and has demonstrated improvements in recognition performance.
Index Terms: Language model combination, WFST

1. Introduction

In automatic speech recognition systems, it is important for the language model to cover the topics or domains of the input utterances. A single accurate language model can be trained if a corpus exists that has similar properties to the inputs. However, such an in-domain corpus is often hard to collect. Moreover, out-of-domain utterances often occur in real applications. An alternative method to cover a wide range of domains, is to utilize a combination of multiple language models.

Linear interpolation is one method to combine multiple language models. It works effectively on topic adaptation using a mixture of topic dependent models [1]. It is used for task switching on mobile speech input [2]. However, the accuracy of the combined model degrades when one of the component models doesn't output accurate probability for target words because it always takes a weighted sum of output probabilities for each component model as follows:

$$P_{LI}(W) = \sum_{i=1}^{M} \lambda_i P_i(W), \text{ where } \sum_{i=1}^{M} \lambda_i = 1. \qquad (1)$$

Here, we consider a word sequence W, which is composed of words in the target domain and is covered by one of M components. Then, the in-domain model could output an accurate probability of W, however, the other models might output unreliable probabilities that are approximated by a back-off procedure or estimated by a small number of training samples. Thus, such a fully combined model tends to output inaccurate probability for W. Such a case occurs when the target domain is broader than each component (e.g. open-domain speech input) and this affects speech recognition performance in an adverse way.

In this paper, we propose a language model combination method that employs a state merging technique for WFSTs to cover a wide range of domains more precisely. The proposed method partially combines components and builds a model we call a *tied-state mixture language model*. We utilize a two-step approach to unite the components. First, we use the union operation to incorporate all of the components into one transducer. This WFST covers a wide range of domains by combining several models of different domains in parallel [3]. Then, we merge a subset of states in the union WFST to mix n-grams from the multiple components and retain unique n-grams in each model simultaneously. Our method is expected to avoid degradation by the interpolation described above by mixing probabilities only for expression common in components.

This paper is organized as follows. Section 2 presents WFST representation of the language model and our proposed method, WFST state merging; and Section 3 describes experimental evaluation results for speech recognition in a travel conversation task.

2. Proposed Method

Our proposed method combines language models with a two-step approach. First, it uses the union operation to incorporate all components into a single WFST. Second, it merges a subset of states in the union WFST.

2.1. WFST Representation of Language Model

The WFST framework for ASR is described in [4]. A WFST is a finite state machine that is able to represent various probabilistic models in ASR such as HMMs for acoustic models or n-grams for language models. It also provides a set of operations to combine and optimize

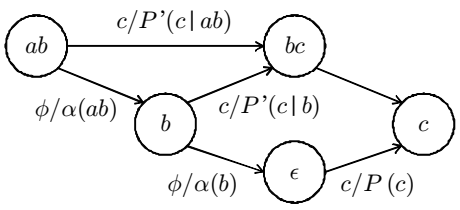

Figure 1: WFST representation of word trigram $P(c|ab)$.

transducers. The decoding process works efficiently, because these operations are applied in advance.

In word n-gram language models, an output probability of word w occurring after word sequence h is expressed as follows:

$$P(w|h) = \begin{cases} \tilde{P}(w|h) & \text{if } c(hw) > 0, \\ \alpha(h)P(w|h') & \text{otherwise.} \end{cases} \quad (2)$$

where $c()$ is a frequency count of word sequence, $\tilde{P}()$ is an n-gram probability adjusted by discounting, and $\alpha()$ is a weight coefficient for the back-off process.

Word n-gram models are naturally represented by WFST [5]. Each state of the WFST corresponds to the history word sequence of n-gram. Each transition has a label and weight encoding output word and probability respectively. For instance, an n-gram probability $P(w|h)$ is expressed as a transition with label w and weight $P(w|h)$ from state h. The back-off procedure in Eq. (2) is described using a failure transition. Figure 1 shows an example of a word trigram represented by WFST.

2.2. Union Language Model

In the union combination, the output probability of word sequence W is calculated using the *sum* operation of WFST as follows:

$$LM_{union}(W) = \bigoplus_i LM_i(W). \quad (3)$$

The exact behaviour of the operation depends on the type of semiring. In the case of a tropical semiring, for instance, the operation corresponds to a minimum function that selects the smallest value.

The structure of a union WFST is interpreted as a parallel connection of components as depicted in Figure 2 (a). A union WFST G_u of component WFSTs G_i consists of a set of states $Q_u = \bigcup_i (Q_i) \cup \{S_u\}$ and a set of transitions $E_u = \bigcup_i (E_i \cup \{\pi(S_u, S_i)\})$, where S_u and S_i are the initial state of G_u and G_i, $\pi(S_u, S_i)$ is the transition from S_u to S_i.

In the union combination, each output probability of a component is calculated individually and united with respect to each utterance unit. It can be interpreted as sentence-level interpolation [6] when using log semiring.

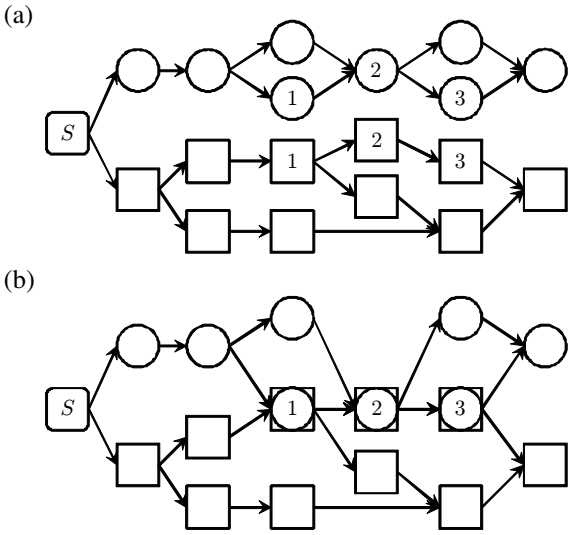

Figure 2: Example of (a) union WFST and (b) tied-state mixture WFST. (a) Two WFSTs G_1 (circle) and G_2 (square) are connected in parallel. (b) The pairs of labeled states $(1, 2, 3)$ of union WFST are merged.

2.3. Tied-State Mixture Language Model

Our proposed method mixes component language models by a state merging technique. The procedure of state merging is divided into three steps as follows:

1. Select a subset of states to merge.
2. Modify the structure of the WFST.
3. Set weights of any new transitions.

In the explanation below, we desribe how to combine two WFSTs representing different language models G_1 and G_2 to create tied-state mixture language model G_t. First, G_t is initialized by taking the union of G_1 and G_2. Note that the following steps are also applicable when there are more than two components.

Step 1: We select a subset of states Q_m from Q_t. Here, we merge states corresponding to n-grams which have a common history in the respective components. To do this, only when multiple states in Q_t represent the same history, we add them to Q_m. In our example, the subset is updated by $Q_m = Q_m \cup \{h_1, h_2\}$ for all $\{h_1, h_2\} \in Q_t$, where $h_i \in Q_i$ denotes a state corresponding to history h in the i-th n-gram language model. When only either h_1 or h_2 exists we don't add it to Q_m.

Step 2: We modify the structure of G_t related to the selected states Q_m. Figure 2 (b) shows an example of WFST to which this technique is applied. For each history h included in Q_m, we add a new merged state h_t to Q_t and add new transitions $\{\pi(q, h_t)|q \in \cup_i\, p(h_i)\}$ and $\{\pi(h_t, q)|q \in \cup_i\, n(h_i)\}$ to E_t, where $p()$ and $n()$ are the set of preceding and following states. Then, the states h_1, h_2 and their connecting transitions are removed from Q_t and E_t.

Step 3: We set the weights of transitions created in the previous step. A new probability distribution $P(w|h)$

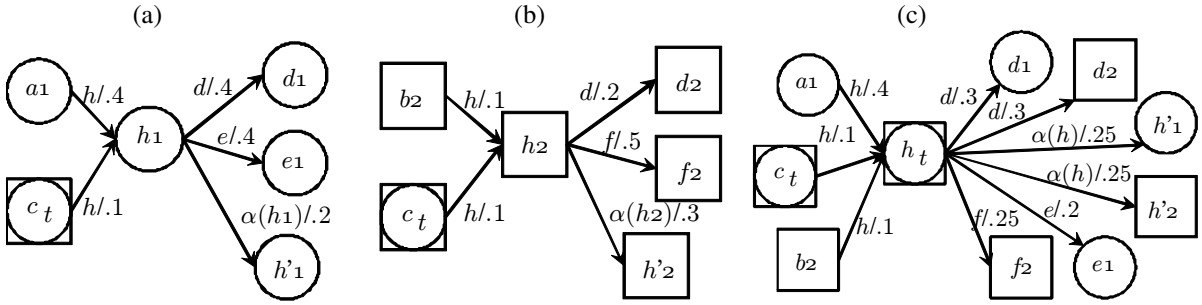

Figure 3: Example of states in WFST. (a) state h_1 and (b) state h_2 are merged into (c) state h_t. Here, weights of transitions leaving h_t are given by $G_t(LI)$ in Table 1. The state c_t is already merged, and d_1 and d_2 are not merged yet at this moment.

Table 1: Example of weights of transitions from the merged state h_t in Figure 3. G_t(LI) is based on Eq. (4) with $\lambda_i = 0.5$, G_t(max) is based on Eq. (5).

	d	e	f	$\alpha(h)$
G_1	0.4	0.4	—	0.2
G_2	0.2	—	0.5	0.3
G_t(LI)	0.3	0.2	0.25	0.25
G_t(max)	0.25	0.25	0.3125	0.1875

of the tied-state mixture language model is represented by the weights of transitions leaving a merged state h_t. We calculate such new weights from the weights of component models. Here, in probability domain, we use the linear interpolation or the maximum selection as follows:

$$P_{\mathrm{LI}}(w|h) = \sum_i \lambda_i P_i(w|h), \quad (4)$$

$$P_{\max}(w|h) = \frac{\max_i P_i(w|h)}{\sum_{\bar{w}} \max_i P_i(\bar{w}|h)}. \quad (5)$$

Table 1 shows an example of weight values corresponding to probabilities $P(w|h)$ $(w \in \{d, e, f\})$ and a back-off coefficient $\alpha(h)$ in Figure 3. These weights sum to one. In this method, the back-off weight is treated in the same way as word probabilities. When a number of transitions have the same label, they have an equivalent weight. In Figure 3, both transition $\pi(h_t, d_1)$ and $\pi(h_t, d_2)$ have the same weight corresponding to a new probability $P(d|h)$.

In addition, for all transitions leading into the merged state, we use their original weight. For example in Figure 3, a weight of $\pi(a_1, h_t)$ is the weight of $\pi(a_1, h_1)$.

The proposed language model has the following properties. (a) It can accept more patterns of word sequences than the union model does because it enables a transition across component models via merged states. It is equivalent to the union model when no states are merged. (b) It outputs an interpolated probability for n-gram of which history is common in components, when using Eq. (4). It can realize word-level interpolation when all states are merged. It outputs an probability deriving from the highest values among component models, when using Eq. (5). (c) It retains the original probability for n-gram included in a single model to avoid the probability degradation caused by the conventional interpolation.

3. Evaluation

3.1. Experimental Setup

We experimentally evaluated the performance of the proposed method on a travel conversation recognition task. The speech data used for the evaluation was a collection of utterances spoken in Japanese. To record the utterances of users in a spontaneous speech style, we used our iPhone application VoiceTra [1], which is available to the public. Although VoiceTra was developed as a client of a multilingual speech translation system on travel conversation tasks, users often input out-of-domain utterances. We sampled 562 utterances randomly for a test set, which contained such utterances.

We used a word trigram for each language model. Two text corpora were used for training the language models. One is the Basic Travel Expression Corpus [7] (BTEC), which is a collection of spoken dialogue data that contains basic travel expressions. The other is a collection of utterances recorded with VoiceTra (VTlog), excluding the test set. The number of words for each corpus was 7.6 million and 1.6 million respectively. Each trigram was built using the SRILM toolkit [8] with modified Kneser-Ney smoothing. The vocabulary size and out-of-vocabulary rate was 64,494 (3.3%) for BTEC and 20,643 (6.1%) for VTlog.

For comparison, we developed three types of mixture models of BTEC and VTlog. We applied word-level linear interpolation and union operation to them with weights optimized on a development set (1450 utterances) excluding the test set. We also merged n-gram frequency of the corpora and trained a word trigram.

We combined two language models, BTEC and VTlog, using our proposed method. In this experiment, we merged all pairs of bigram states (corresponding to history of trigram) included in both models. The other bigram states, all unigram states and zerogram state were not changed at all. The number of merged pairs was 66,868. The number of bigram states was 653,458 and

[1] http://www.mastar.jp/translation/voicetra-en.html

Table 2: Experimental results: Word error rate (WER).

Language Model	WER(%)
BTEC	25.0
VTlog	23.3
N-gram Merging	23.4
Linear Interpolation	22.9
WFST Union	22.7
Proposed 1	21.9
Proposed 2	21.8

100,709 in the BTEC and VTlog respectively. We calculated new weights with two types of functions: Eq. (4) (Proposed 1) and Eq. (5) (Proposed 2).

For the speech recognition, we used our WFST-based decoder [9], which can compose WFST language models on-the-fly during decoding. The search parameters were configured to achieve $\mathrm{RTF} < 1$. Acoustic models were gender-dependent triphone HMMs (5,670 states) trained with 271 hours speech excluding the test set, and a 25-dimension feature vector (MFCC1-12 and its Δ, Δlog-power) was extracted from each frame.

3.2. Experimental Results

Table 2 shows experimental results for the proposed tied-state mixture model and for conventional mixture models. The word and error rate (WER) with our method (Proposed 1, 2) was lower than that with the other models (N-gram Merging, Linear Interpolation, WFST Union). This result indicates that the proposed method contributed to improving recognition performance. Proposed 1 and 2 performed well on increasing correct results and reducing insertion errors respectively.

From the recognition results we investigated corresponding paths in the search WFST. From this analysis, it was found that the proposed method output hypotheses that transit across the two component models. This was one of reasons our model could outperform the standard union method. We also found that the proposed method correctly recognized utterances that were mis-recognized by the linear interpolation method. For such utterances, the linear interpolation method incorporated low back-off probabilities from the component models and that gave lower scores to the correct paths. Figure 4 shows a part of the proposed model corresponding to a part of the utterance on which our model worked better than the conventional models.

The BTEC model covers utterances of travel domain, and the VTlog model covers users' real utterances of both travel domain and other domains. The n-gram merging could incorporate a unique part of VTlog by combining in frequency domain. But it couldn't achieve enough improvement for travel domain because VTlog was smaller than BTEC. The proposed method provided a language model that is simultaneously accurate for the travel domain by mixing components and the other domains by using each component individually.

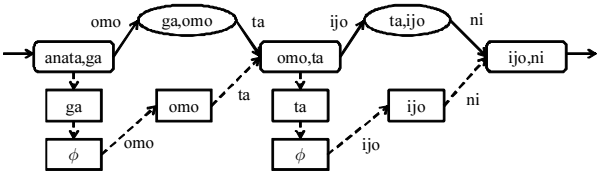

Figure 4: Part of the proposed model for four words "omo, ta, ijo, ni". Our model could trace solid arrows with trigram probabilities. The linear interpolation degraded on these words because it should have taken into account dotted arrows with back-off probabilities.

4. Summary and Future Work

In this paper, we proposed a language model combination method that employs the WFST union operation and a state merging technique to cover a wide range of domains using multiple models. The proposed method first applies a union operation to the component language models and then merges parts of the union transducer to mix n-grams included from the multiple models and to retain unique n-grams from each model simultaneously. The effectiveness of the proposed method was demonstrated in a series of speech recognition experiments using VoiceTra, and our method achieved an improvement in recognition performance.

Future work is to include more experimental evaluations under the condition of combining more models across various domains. We are also considering how to improve our algorithm for selecting states to merge and weighting transitions from the merged state.

5. References

[1] R. Kneser and V. Steinbiss, "On the dynamic adaptation of stochastic LM," in *Proc. of ICASSP*, 1993, pp. 586–589.

[2] B. Ballinger, *et al.*, "On-demand language model interpolation for mobile speech input," in *Proc. of Interspeech*, 2010, pp. 1812–1815.

[3] X. Liu, *et al.*, "Language model combination and adaptation using weighted finite state transducers," in *Proc. of ICASSP*, 2010, pp. 5390–5393.

[4] M. Mohri, *et al.*, "Weighted finite-state transducers in speech recognition," *Computer Speech and Language*, vol. 16, no. 1, pp. 69–88, 2002.

[5] C. Allauzen, *et al.*, "Generalized algorithms for constructing statistical language models," in *Proc. of ACL*, 2003, pp. 40–47.

[6] R. M. Iyer and M. Ostendorf, "Modeling long distance dependence in language: Topic mixtures versus dynamic cache models," *IEEE Trans. on Speech and Audio Processing*, vol.7, no.1, pp.30–39, 1999.

[7] T. Takezawa, *et al.*, "Toward a broad-coverage bilingual corpus for speech translation of travel conversations in the real world," in *Proc. of LREC*, 2002, pp. 147–152.

[8] A. Stolcke, "SRILM - An extensible language modeling toolkit," in *Proc. of ICSLP*, 2002.

[9] P. R. Dixon, *et al.*, "A comparison of dynamic WFST decoding approaches," in *Proc. of ICASSP*, pp. 4209–4212, 2012.