COMPONENT FUSION: LEARNING REPLACEABLE LANGUAGE MODEL COMPONENT FOR END-TO-END SPEECH RECOGNITION SYSTEM

Changhao Shan[1,2*], Chao Weng[4], Guangsen Wang[3], Dan Su[3], Min Luo[2], Dong Yu[4], Lei Xie[1†]

[1]School of Computer Science and Engineering, Northwestern Polytechnical University, Xian, China
[2]Tencent AI Platform Department, Shenzhen, China
[3]Tencent AI Lab, Shenzhen, China
[4]Tencent AI Lab, Bellevue, USA

{chshan, lxie}@nwpu-aslp.org, {cweng, vincegswang, dansu, selwynluo, dyu}@tencent.com

ABSTRACT

Recently, attention-based end-to-end automatic speech recognition system (ASR) has shown promising results. One of the limitations of an attention-based ASR system is that its language model (LM) component has to be implicitly learned from transcribed speech data which prevents one from utilizing plenty of text corpora to improve language modeling. In this work, the Component Fusion method is proposed to incorporate externally trained neural network (NN) LM into an attention-based ASR system. During training stage we equip the attention-based system with an additional LM component which is replaced by an externally trained NN LM at decoding stage. Experimental results show that the proposed Component Fusion outperforms two prior LM fusion approaches, i.e., Shallow Fusion and Cold Fusion, in both out-of-domain and in-domain scenarios. Further improvements can be achieved when combining Component and Shallow Fusion.

Index Terms— automatic speech recognition, end-to-end speech recognition, attention-based model, language model

1. INTRODUCTION

A conventional NN based ASR system [1, 2, 3, 4, 5] includes several individual components such as acoustic model, lexicon and LM, etc. Recently, attention-based end-to-end ASR system consolidates all necessary ASR components into one neural framework and has achieved state-of-the-art results on several speech tasks, such as LVCSR [6, 7, 8, 9], speaker verification [10] and keyword spotting [11]. Unlike a conventional ASR system where an LM can be separately trained on text data, the LM component of an attention-based ASR system has to be learned implicitly from transcribed speech which is usually less than available text corpora by orders of magnitude. To address this, the semi-supervised training method is employed by Karita et al. [12] to exploit large text data by the shared encoder architecture and text-to-text auto-encoder and achieved a better result for end-to-end ASR. The LM fusion [8, 13, 14, 15, 16, 17, 18, 19] is another way to alleviate this issue. According to these approaches, the LM is first externally trained on text data and then incorporated into end-to-end ASR model. Shallow Fusion [8, 16, 17] interpolates the label probabilities with the ones from an external LM during inference stage. For Deep Fusion [14] and Cold Fusion [15], an external NN LM is incorporated into attention-based system through gating mechanism and the learned gating parameters usually lead to a better performance.

In this work, inspired by the Cold Fusion, Component Fusion is proposed to incorporate externally trained NN LM into an attention-based ASR system. During training stage we equip the attention-based system with an additional LM component which can be replaced by any externally trained NN LM at inference stage. As opposed to Deep or Cold Fusion, Component Fusion enables quick domain adaptation by reusing attention-based model whereas replacing a new externally trained in-domain NN LM. Experimental results show that the proposed Component Fusion consistently outperforms Shallow and Cold Fusion, in both out-of-domain and in-domain scenarios. Further improvements can be achieved when combining Component and Shallow Fusion together.

The remainder of the paper is organized as follows: Section 2 presents attention-based ASR model. We then review some existing approaches for incorporating external LM to attention-based systems in Section 3. The proposed Component Fusion is detailed in Section 4. Experimental results are given in Section 5. Section 6 concludes our work.

2. ATTENTION-BASED MODEL

The baseline attention-based end-to-end ASR system is depicted in Figure 1(a). The encoder module results in a high

The research work is supported by the National Key Research and Development Program of China (No.2017YFB1002102) and Tencent AI Lab Rhino-Bird Joint Research Program (No.JR201853).

*Work performed during an internship at Tencent.

†Corresponding author.

level representation $\mathbf{h^{enc}}$ from the input acoustic feature $\mathbf{x}$. Based on the encoder and decoder hidden output, using the content-based attention [20], a context vector c_t is computed from the attention module to form a weighted sum of the encoder outputs. The decoder hidden state output h_t^{dec} is computed from the context vector c_{t-1} and the target label y_{t-1} from the previous time step. Finally, the output y_t is obtained after a projection and softmax layer. The whole computation process can be summarized as follows:

$$\boldsymbol{h^{enc}} = Encoder(\boldsymbol{x}), \tag{1}$$

$$h_t^{dec} = Decoder(y_{t-1}, c_{t-1}), \tag{2}$$

$$c_t = Attend(\boldsymbol{h^{enc}}, h_t^{dec}), \tag{3}$$

$$h_t^{att} = tanh(\boldsymbol{W_h}[c_t; h_t^{dec}]), \tag{4}$$

$$y_t = softmax(\boldsymbol{W_o} h_t^{att}). \tag{5}$$

In this work, we applied some improvements to the baseline system as proposed in [9, 21], including input-feeding and softmax smoothing. Specifically, the context vector c_{t-1} is replaced with the attentional hidden state h_{t-1}^{att} as the inputs to the decoder when computing the decoder hidden state as given in equation (6). In addition, softmax smoothing [8, 22] is used to smooth the network label prediction distribution during decoding as given in equation (7). The temperature hyperparameter $\tau = 2$ is to control the smoothing strength.

$$h_t^{dec} = Decoder(y_{t-1}, h_{t-1}^{att}), \tag{6}$$

$$y_t = softmax(\boldsymbol{W_o} h_t^{att}/\tau). \tag{7}$$

3. RELATED WORK

In this section, we review some existing approaches of incorporating external LMs trained from large text corpora to the attention-based end-to-end systems, including Shallow Fusion, Deep Fusion and Cold Fusion.

3.1. Shallow Fusion

For Shallow Fusion, the decoding score $logP(y_t)$ is computed as equation (8), where $logP(y_t)$ is the log posterior produced by the network. $logP_{LM}(y_t)$ and $P_{Att}(y_t)$ are the output of the external LM and attention-based model respectively. β is a tunable parameter for weighting the external LM score. In this work, character-based RNN language [16] model is used to provide the external LM score. As depicted in Figure 1(b), the interpolated score $logP(y_t)$ is the used for decoding based on a simple left-to-right beam search algorithm [7]. It is also worth noting that the external RNN LM is trained independently of the attention-based system and fused with the attention-based model only at decoding stage.

$$logP(y_t) = logP_{Att}(y_t) + \beta logP_{LM}(y_t), \tag{8}$$

.

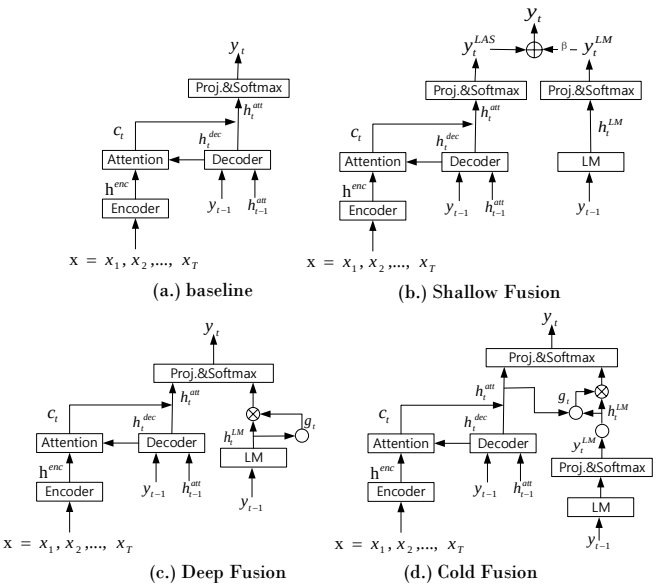

Fig. 1. A schematic representation of various LM fusion approaches.

3.2. Deep Fusion

Deep Fusion was firstly proposed in [14] for machine translation and subsequently successfully used for ASR in [15]. As illustrated in Figure 1(c.), the external LM is incorporated in Deep Fusion by concatenating the hidden state outputs of both the attention-based end-to-end ASR system and the pre-trained LM:

$$g_t = sigmoid(\boldsymbol{U_g} s_t^{LM} + b), \tag{9}$$

$$\widehat{h}_t^{att} = [h_t^{att}; g_t s_t^{LM}], \tag{10}$$

$$y_t = softmax(\boldsymbol{W_o'} \widehat{h}_t^{att}), \tag{11}$$

where g_t is a gate output parameterized by U_g controlling the importance of the contribution of the hidden state of LM s_t^{LM}. The concatenated hidden state output $\widehat{h}_t^{att}$ is then used to predict the target label through the softmax function parameterized by $\boldsymbol{W}_o^i$. Note that for Deep Fusion, the LM parameters are also trained independently from the end-to-end ASR system. Finally, the combining parameters $\boldsymbol{U_g}, \boldsymbol{W_o'}$ are fine-tuned on a small amount of data.

3.3. Cold Fusion

In Cold Fusion (Figure 1(d.)), the LM and the attention-based model components are combined at the projection and softmax layer. Code Fusion explores the LM information even further by training the end-to-end ASR model from scratch jointly with a fixed pre-trained LM. The motivation is to retain only the relevant language information for mapping from the source to the target sequence. The Cold fusion can be

formulated as follows:

$$h_t^{LM} = DNN(l_t^{LM}), \tag{12}$$

$$g_t = sigmoid(\boldsymbol{U_g}[h_t^{LM}; h_t^{att}] + b), \tag{13}$$

$$\widehat{h}_t^{att} = [h_t^{att}; g_t h_t^{LM}], \tag{14}$$

$$y_t = softmax(\boldsymbol{W_o'}\widehat{h}_t^{att}). \tag{15}$$

where l_t^{LM} is the logit output of the external LM in [15] and $[\cdot;\cdot]$ denotes the concatenation of two vectors. In our work, the l_t^{LM} is replaced with the output of the LM y_t^{LM} for better convergence. And the DNN is a projection layer of 1,024 units.

4. COMPONENT FUSION

In this section, we describe the proposed Component Fusion approach. For the conventional end-to-end ASR systems, the trained system is highly biased towards the training data domain due to the LM component, i.e., the decoder is trained only with the transcribed speech of limited size. Inspired by Cold Fusion, for Component Fusion, we proposed to "decouple" the language modeling component from the system training by making the external LM component a replaceable one in Cold Fusion.

Unlike Cold Fusion, the external LM component is trained on speech transcriptions instead of a larger external text corpus. This is to reduce the mismatch between the external LM and the decoder component of the end-to-end system so that replacing the external LM may have a similar effect of removing the LM component of the end-to-end system, i.e., decoupling the acoustic and LMs of the end-to-end system. Another advantage of this is that the external LM is fast to converge and performs better on the training data than if it was trained with larger external text corpora.

This training scheme offers two major advantages. Firstly, during decoding, we can replace the external LM component with the new one trained with text corpora with a much larger scale to improve the recognition performance. Furthermore, we can even use an LM trained with a totally different domain to replace the external LM component for fast domain adaptation without retraining the whole system.

Apart from the training strategy difference from Cold Fusion, as illustrated in Figure 2, another modification has been made to let the LM impact the training of the ASR system in an earlier stage. This is achieved by concatenating the gated LM output with the output of the decoder h^{dec} rather than the attentional output as in equation (14) for the Cold Fusion:

$$h_t^{dec} = [h_t^{dec}; g_t h_t^{LM}]. \tag{16}$$

5. EXPERIMENTS

We examined the proposed Component Fusion in two scenarios, out-of-domain and in-domain. For out-of-domain, i.e., domain adaptation, we collected two datasets: the monolingual Mandarin and English dataset which served as source data, the Mandarin-English code-switching dataset which served as target data. For the in-domain case, a RNN LM was trained on a considerably larger amount of external in-domain text and incorporated into the attention-based system using Component Fusion. The public data sets, AISHELL-1 and AISHELL-2 [23], were used. We adopt similar setups between the two scenarios as described below.

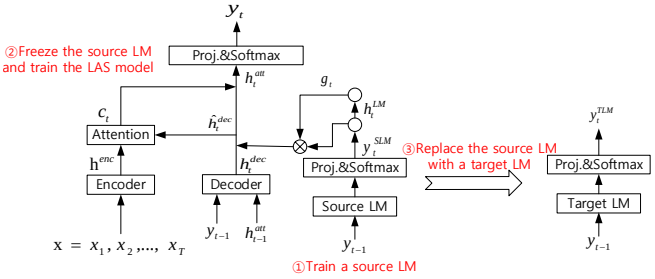

Fig. 2. The Component Fusion that concatenates the gated LM with the hidden state of the decoder h^{dec}.

We built an attention-based end-to-end speech recognition system. The encoder consists of 6 BLSTM layers each with 1024 LSTM units. And the decoder is a 2 LSTM layers each with 1024 LSTM units. We employed the ADAM algorithm [24] with default parameters and the initial learning rate is set to 0.001. We also halve the learning rate if there is no improvement on the validation set. Meanwhile, we apply dropout [25] with probability 0.2 during training to reduce overfitting. We used characters as the target labels which include English letters, Mandarin characters, punctuations plus '<space>', '<SOS>' and '<EOS>'. Besides, each audio frame was computed based on a 80-channel Mel-filterbank with 25ms windowing and 10ms frame shift. Mean and variance normalization was conducted for each speaker.

For the external LM used in all experiments, we adopted a character-based 3-layer GRU model each with 1024 GRU units and used the similar training procedure as in attention model described above. Meanwhile, we used characters as the target labels.

5.1. Out-of-domain

For the domain adaptation, we collected a dataset which contains monolingual Mandarin and English data as the source domain. The dataset has about 1K hours (about 1M utterances) Mandarin speech data and about 100 hours (about 110K utterances) English speech data. Besides, Mandarin-English code-switching was served as the target domain and we collected about 12.4 hours code-switching speech data as the test dataset. We also collected about 810K code-switching sentences data for the LM. For fast training and convergence, we spliced the central frame with left 3 plus right 3 frames

Table 1. *Perplexities (PPL) on the code-switching test set.*

Data	PPL
source	63.02
target	13.12
source + target	12.46

Table 2. *Performance for the out-of-domain scenarios.*

Model	CER (%)
baseline	28.33
+ Shallow Fusion	25.01
Cold Fusion (h^{att})	25.37
Cold Fusion (h^{dec})	21.54
+ Shallow Fusion	20.40
Component Fusion (h^{att})	20.43
Component Fusion (h^{dec})	**17.68**
+ Shallow Fusion	**17.53**

and subsample the input by a factor of three.

Table 1 shows the perplexities (PPL) of different domain NN LMs on code-switching test set. Note that since our NN LM was built on character level, the perplexities are lower than those build on word level in general. We can clearly see that there exists significant mismatch between the source and target domain. Domain adaptation is performed using the proposed Component Fusion: 1. the $source$ NN LM was trained on speech transcriptions and then frozen when attention model is being trained. 2. during decoding, the $source$ NN LM is replaced by the $target$ NN LM trained on target domain. We also explored the combination of Component Fusion and Shallow Fusion at decoding stage.

In Table 2, we can observe that incorporating the LM into attention-based model achieves a great performance. The Shallow Fusion already achieves a significant improvement. Compared to Shallow Fusion, the Cold Fusion with h^{att} achieves a similar results and our Component Fusion presents a better performance. In addition, as described in Section 4, we explore the performance of concatenating the gated LM with the hidden state of the encoder h^{dec}. This means the LM can impact the training of the attention-based model in an earlier stage. Table 2 shows that both Component and Cold Fusion with h^{dec} achieve a better performance. Finally, using the Shallow Fusion, the Component and Cold Fusion can further improve the performance.

5.2. In-domain

For the in-domain case, we evaluated our model on the AISHELL-1 dataset [23]. The AISHELL-1 dataset contains 11 domains, including the smart home, autonomous driving, and industrial production. And the dataset consists of about 178 hours Mandarin speech data, which has $\sim$120K utterances. On the AISHELL-1 dataset, the test set has 7,176 (about 5 hours) utterances. Meanwhile, we selected about 529K sentences from AISHELL-2 text data to train a better LM. Different from code-switching task, we concatenated the Mel-filterbank feature, deltas and delta-deltas as they led to a better performance on AISHELL-1 sets.

Table 3. *Perplexities (PPL) on the AISHELL-1 test set.*

Data	PPL
AISHELL-1	41.80
AISHELL-2	29.68

Table 4. *Performance on the AISHELL-1 dataset.*

Model	CER (%)
baseline	10.56
+ Shallow Fusion	9.78
Cold Fusion (h^{att})	9.31
+ Shallow Fusion	8.77
Cold Fusion (h^{dec})	10.10
Component Fusion (h^{att})	**9.04**
+ Shallow Fusion	**8.71**
Component Fusion (h^{dec})	10.26

From Table 3, not surprisingly, a large text corpora achieved a better performance. For Component Fusion, the $AISHELL$-1 NN LM was first trained on AISHELL-1 speech transcriptions. Then the attention model is trained with $AISHELL$-1 NN LM fixed before it is replaced by the $AISHELL$-2 NN LM during decoding.

Similar to out-of-domain, we also notice that the external LM can lead to a significant improvement. The Shallow Fusion achieves a better result than baseline. Besides, both Component and Cold Fusion with h^{att} are preferred over Shallow Fusion and Component Fusion achieves the best performance. Different from out-of-domain, combining the LM with h^{dec} achieves a worse performance. We analyzed that it is unwise to interfere the training of attention-based model in an earlier stage for the in-domain case. The training data collected from the same domain are sufficient to learn a good attention-based model and incorporating the LM with label prediction h^{att} can further improve the model's performance. In addition, for Component and Cold Fusion, we continue to observe the gains by using the Shallow Fusion.

We explored the performance of incorporating the LM into attention-based model in two scenarios. For in-domain and out-of-domain, we obtain the same conclusions that the external LM can improve the performance and the Component Fusion consistently achieves the best result.

6. CONCLUSION

In this work, we propose the Component Fusion to incorporate an external LM into attention-based model. The proposed method allows both exploiting large text training corpora and fast domain adaptation for an attention-based end-to-end ASR system. We evaluate Component Fusion on two scenarios including out-of-domain and in-domain. In both scenarios, Component Fusion consistently outperformed Deep Fusion and Cold Fusion. Further improvements were achieved when combining Component Fusion with Shallow Fusion.

7. REFERENCES

[1] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal processing magazine*, 2012.

[2] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Transactions on audio, speech, and language processing*, 2012.

[3] L. Deng, J. Li, J.-T. Huang, K. Yao, D. Yu, F. Seide, M. L. Seltzer, G. Zweig, X. He, J. D. Williams *et al.*, "Recent advances in deep learning for speech research at microsoft." in *ICASSP2013*.

[4] L. Deng, D. Yu *et al.*, "Deep learning: methods and applications," *Foundations and Trends® in Signal Processing*, vol. 7, no. 3–4, pp. 197–387, 2014.

[5] S. Sun, B. Zhang, L. Xie, and Y. Zhang, "An unsupervised deep domain adaptation approach for robust speech recognition," *Neurocomputing*, vol. 257, pp. 79–87, 2017.

[6] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *ICASSP2016*.

[7] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Advances in neural information processing systems*, 2015, pp. 577–585.

[8] C. Shan, J. Zhang, Y. Wang, and L. Xie, "Attention-based end-to-end speech recognition on voice search," in *ICASSP2018*.

[9] C. Weng, J. Cui, G. Wang, J. Wang, C. Yu, D. Su, and D. Yu, "Improving attention based sequence-to-sequence models for end-to-end english conversational speech recognition," in *Interspeech2018*.

[10] F. Chowdhury, Q. Wang, I. L. Moreno, and L. Wan, "Attention-based models for text-dependent speaker verification," *arXiv preprint arXiv:1710.10470*, 2017.

[11] C. Shan, J. Zhang, Y. Wang, and L. Xie, "Attention-based end-to-end models for small-footprint keyword spotting," in *Interspeech2018*.

[12] S. Karita, S. Watanabe, T. Iwata, A. Ogawa, and M. Delcroix, "Semi-supervised end-to-end speech recognition," in *Interspeech2018*.

[13] T. Mikolov, M. Karafiát, L. Burget, J. Černocký, and S. Khudanpur, "Recurrent neural network based language model," in *Interspeech2010*.

[14] C. Gulcehre, O. Firat, K. Xu, K. Cho, L. Barrault, H.-C. Lin, F. Bougares, H. Schwenk, and Y. Bengio, "On using monolingual corpora in neural machine translation," *arXiv preprint arXiv:1503.03535*, 2015.

[15] A. Sriram, H. Jun, S. Satheesh, and A. Coates, "Cold fusion: Training seq2seq models together with language models," in *Interspeech2018*.

[16] T. Hori, S. Watanabe, and J. R. Hershey, "Multi-level language modeling and decoding for open vocabulary end-to-end speech recognition," in *ASRU2017*.

[17] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," in *ICASSP2016*.

[18] S. Toshniwal, A. Kannan, C.-C. Chiu, Y. Wu, T. N. Sainath, and K. Livescu, "A comparison of techniques for language model integration in encoder-decoder speech recognition," *arXiv preprint arXiv:1807.10857*, 2018.

[19] Z. Chen, M. Jain, Y. Wang, M. L. Seltzer, and C. Fuegen, "End-to-end contextual speech recognition using class language models and a token passing decoder."

[20] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[21] T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *EMNLP2015*.

[22] J. Chorowski and N. Jaitly, "Towards better decoding and language model integration in sequence to sequence models," *arXiv preprint arXiv:1612.02695*, 2016.

[23] H. Bu, J. Du, X. Na, B. Wu, and H. Zheng, "Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline," in *O-COCOSDA2017*.

[24] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[25] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.