Interspeech 2018
2-6 September 2018, Hyderabad

Cold Fusion: Training Seq2Seq Models Together with Language Models

Anuroop Sriram, Heewoo Jun, Sanjeev Satheesh & Adam Coates

Baidu Research
Sunnyvale, USA

{sriramanuroop,junheewoo,sanjeevsatheesh,acoates}@baidu.com

Abstract

Sequence-to-sequence (Seq2Seq) models with attention have excelled at tasks which involve generating natural language sentences such as machine translation, image captioning and speech recognition. Performance has further been improved by leveraging unlabeled data, often in the form of a language model. In this work, we present the Cold Fusion method, which leverages a pre-trained language model **during training**, and show its effectiveness on the speech recognition task. We show that Seq2Seq models with Cold Fusion are able to better utilize language information enjoying i) faster convergence and better generalization, and ii) almost complete transfer to a new domain while using less than 10% of the labeled training data.

1. Introduction

Sequence-to-sequence (Seq2Seq) [1] models have achieved state-of-the-art results on many natural language processing problems including automatic speech recognition [2, 3] and neural machine translation [4]. With a sufficiently large labeled dataset, vanilla Seq2Seq can model sequential mapping well, but it is often augmented with a language model to further improve the fluency of the generated text.

The standard way to integrate language models is to train the Seq2Seq model and the language model independently and then combine their outputs to guide beam search [5, 6, 7].

While these approaches have been shown to improve performance over the baseline, they have a few limitations. First, because the Seq2Seq model is trained to output complete label sequences without a language model, its decoder learns an implicit language model from the training labels, taking up a significant portion of the decoder capacity to learn redundant information. Second, the residual language model baked into the Seq2Seq decoder is biased towards the training labels of the parallel corpus. Thus, in order to adapt to novel domains, the Seq2Seq model must first learn to discount the implicit knowledge of the language.

In this work, we introduce *Cold Fusion* to overcome both these limitations. Cold Fusion encourages the Seq2Seq decoder to learn to use the external language model during training. This means that Seq2Seq can naturally leverage potentially limitless unsupervised text data, making it proficient at adapting to a new domain. The latter is especially important in practice as the domain from which the model is trained can be different from the real world use case for which it is deployed. In our experiments, Cold Fusion can almost completely transfer to a new domain for the speech recognition task with 10 times less data. Additionally, the decoder only needs to learn task relevant information, and thus trains faster.

2. Background and Related work

A basic Seq2Seq model comprises an encoder that maps an input sequence $\mathbf{x} = (x_1, \ldots, x_T)$ into an intermediate representation $\mathbf{h}$, and a decoder that in turn generates an output sequence $\mathbf{y} = (y_1, \ldots, y_K)$ from $\mathbf{h}$ [8]. The decoder can also attend to a certain part of the encoder states with an attention mechanism. For the automatic speech recognition (ASR) task, the Seq2Seq model is called an *acoustic model* (AM) and maps a sequence of spectrogram features extracted from a speech signal to characters.

During inference, we aim to compute the most likely sequence $\hat{\mathbf{y}} = \underset{\mathbf{y}}{\operatorname{argmax}} \log p(\mathbf{y}|\mathbf{x})$ where $p(\mathbf{y}|\mathbf{x})$ is the probability that the task-specific Seq2Seq model assigns to sequence $\mathbf{y}$ given input sequence $\mathbf{x}$. The argmax operation is intractable in practice so we use a left-to-right beam search algorithm similar to the one presented in [6].

A standard way to integrate the language model with the Seq2Seq decoder is to change the inference task to: $\hat{\mathbf{y}} = \underset{\mathbf{y}}{\operatorname{argmax}} \log p(\mathbf{y}|\mathbf{x}) + \lambda \log p_{\mathrm{LM}}(\mathbf{y})$, where $p_{\mathrm{LM}}(\mathbf{y})$ is the language model probability assigned to the label sequence $\mathbf{y}$. [5, 4] describe several heuristics that can be used to improve this basic algorithm. We refer to all of these methods collectively as ***Shallow Fusion***, since p_{LM} is only used during inference.

[7] proposed ***Deep Fusion*** for machine translation that tightens the connection between the decoder and the language model by combining their states with a parametric gating. In Deep Fusion, the Seq2Seq model and the language model are first trained independently and later combined using a learned gate. The parameters of the gate are trained on a small amount of data keeping the rest of the model fixed, and allow the gate to decide how important each of the models are for the current time step.

The biggest disadvantage with Deep Fusion is that the task-specific model is trained independently from the language model. This means that the Seq2Seq decoder needs to learn a language model from the training data labels, which can be rather parsimonious compared to the large text corpora available for language model training. So, the fusion mechanism should learn to overcome this bias in order to incorporate the new language information. This also means that a considerable portion of the decoder capacity is wasted.

3. Cold Fusion

Our proposed Cold Fusion method is largely motivated from the Deep Fusion idea but with some important differences. The biggest difference is that in Cold Fusion, the Seq2Seq model is trained from scratch together with a fixed pre-trained language model. Because the Seq2Seq model is aware of the language model throughout training, it learns to use the language model for language specific information and capture only the relevant

10.21437/Interspeech.2018-1392

information conducive to mapping from the source to the target sequence. This disentanglement can increase the effective capacity of the model significantly. This effect is demonstrated empirically in Section 4 where Cold Fusion models perform well even with a very small decoder.

We also improve on some of the modeling choices of the fusion mechanism.

1. First, both the Seq2Seq hidden state s_t and the language model hidden state s_t^{LM} can be used as inputs to the gate computation. The task-specific model's embedding contains information about the encoder states which allows the fused layer to decide its reliance on the language model in case of input uncertainty. For example, when the input speech is noisy or a token unseen by the Seq2Seq model is presented, the fusion mechanism learns to pay more attention to the language model.
2. Second, we employ fine-grained (FG) gating mechanism [9]. By using a different gate value for each hidden node of the language model's state, we allow for greater flexibility in integrating the language model because the fusion algorithm can choose which aspects of the language model it needs to emphasize more at each time step.
3. Third, we replace the language model's hidden state with the language model probability. The distribution and dynamics of s_t^{LM} can vary considerably across different language models and data. By projecting the token distribution onto a common embedding space, LMs that model novel uses of the language can still be integrated without state discrepancy issues. This also means that we can train or swap with n-gram LMs during inference.

The *Cold Fusion* layer works as follows:

$$h_t^{\text{LM}} = \text{DNN}(\ell_t^{\text{LM}}) \tag{1a}$$

$$g_t = \sigma(W[s_t; h_t^{\text{LM}}] + b) \tag{1b}$$

$$s_t^{\text{CF}} = [s_t; g_t \circ h_t^{\text{LM}}] \tag{1c}$$

$$r_t^{\text{CF}} = \text{DNN}(s_t^{\text{CF}}) \tag{1d}$$

$$\hat{P}(y_t|x, y_{<t}) = \text{softmax}(r_t^{\text{CF}}) \tag{1e}$$

ℓ_t^{LM} is the logit output of the language model, s_t is the state of the task specific model, and s_t^{CF} is the final fused state used to generate the output. Since logits can have arbitrary offsets, the maximum value is subtracted off before feeding into the layer. In (1a), (1d), the DNN can be a deep neural network with any number of layers. In out experiments, DNN in (1a) is a linear map. In (1d), we used two affine layers with ReLU nonlinearity in the hidden layer. Disabling the gate in (1c) reduces this architecture to a vanilla Seq2Seq.

4. Experiments

4.1. Setup

We tested the Cold Fusion method on the speech recognition task. For language model integration experiments on a single domain, we used the publicly available LibriSpeech dataset [10]. It comprises 960 hours of public domain audio books and provides a 800-million-word corpus curated from 14500 books. The language model for this fusion task was one layer of 1536 dimensional gated recurrent units (GRU) [11]. It was trained on the entire LibriSpeech LM corpus with Adam [12] by minimizing the cross-entropy of predicting the next character given the past character sequence. This model obtains 2.617 perplexity on the development set annotations of the LibriSpeech ASR corpus, whereas a 16-gram character LM gets 2.775.

For domain transfer experiments, we collected two data sets: one based on search queries which served as our *source* domain, and another based on movie transcripts which served as our *target* domain. For each dataset, we used Amazon Mechanical Turk to collect audio recordings of speakers reading out the text. The source dataset contains 411,000 utterances (about 650 hours of audio), and the target dataset contains 345,000 utterances (about 676 hours of audio). We held out 2048 utterances chosen uniformly at random from each domain for evaluation. The text of the two datasets differ significantly. A character language model trained on the source domain gets a perplexity of 2.670 on the source domain and a perplexity of 4.463 on the target domain indicating the difference between the source and target domains.

The language model that we used for domain transfer was trained on about 25 million words. This model contained three layers of gated recurrent units (GRU) [11] with a hidden state dimension of 1024. We used the Adam optimizer [12] with a batch size of 512. The model gets a perplexity of 2.49 on the source data and 2.325 on the target data.

For the acoustic models, we used the Seq2Seq architecture with soft attention based on [2]. The encoder consists of 6 bidirectional LSTM (BLSTM) [15] layers each with a dimension of 512 for LibriSpeech experiments and 480 for domain transfer experiments. We also use max pooling layers with a stride of 2 along the time dimension after the first two BLSTM layers, and add residual connections [16] for each of the BLSTM layers to help speed up the training process. The decoder consisted of a single layer of 960 dimensional gated recurrent unit (GRU) with a hybrid attention [17]. The final Cold Fusion mechanism had one dense layer of 256 units followed by ReLU before softmax.

The input sequence consisted of 40 mel-scale filter bank features. We expanded the datasets with noise augmentation; a random background noise is added with a 40% probability at a uniform random SNR between 0 and 15 dB. We did not use any other form of regularization.

We trained the entire system end-to-end with Adam [12] with a batch size of 64. The learning rates were tuned separately for each model using random search. To stabilize training early on, the training examples were sorted by increasing input sequence length in the first epoch [14]. During inference, we used beam search with a fixed beam size of 128 for all of our experiments. We used coverage and length penalties as described in [18] for improved performance during beam search. We also used scheduled sampling [19] with a sampling rate of 0.2 which was kept fixed throughout training.

4.2. Improved Generalization

Leveraging a language model that achieves a low perplexity on the distribution of interest should directly mean an improved WER for the ASR task. In this section, we compare how the different fusion methods fare in achieving this effect on LibriSpeech.

We first trained a baseline attention model that has the same architecture as described in setup with 512 hidden state dimensions for all recurrent cells. As shown in Table 1, this already obtains competitive word error rates on various test sets, and is comparable to Wav2Letter [13] with the power spectrum input features like ours and shallow fusion decoding with a 4-gram word language model. This suggests that the attention decoder

Model	LibriSpeech Test-Clean		LibriSpeech Test-Other		Target Domain Test	
	CER	WER	CER	WER	CER	WER
Wav2Letter + Shallow Fusion						
MFCC	6.9%	7.2%				
Power Spectrum	9.1%	9.4%				
Raw Wave	10.6%	10.1%				
Baseline Attention	4.47%	8.94%	11.57%	21.52%	20.06%	35.35%
Baseline + Deep Fusion	5.01%	9.17%	12.67%	21.48%	22.07%	35.70%
Baseline + Cold Fusion	**3.87%**	**7.47%**	**9.28%**	**17.05%**	**18.29%**	**31.88%**

Table 1: *Results from models trained on the publicly available Librispeech data. Results from the Wav2Letter model [13] are presented for reference*

Model	Train Domain	Source Domain Test		Target Domain Test		
		CER	WER	CER	WER	Domain Gap
Baseline CTC Model + Shallow Fusion	Source	8.38%	14.85%	18.92%	47.46%	
Baseline Attention Model	Source	7.54%	14.68%	23.02%	43.52%	100%
Baseline Attention Model	Target			8.84%	17.61%	0%
Baseline + Deep Fusion	Source	7.64%	13.92%	22.14%	37.45%	76.57%
+ s^{AM} in gate	Source	7.61%	13.92%	21.07%	37.9%	78.31%
+ Fine-Grained Gating	Source	7.47%	13.61%	20.29%	36.69%	73.64%
+ ReLU layer	Source	7.50%	13.54%	21.18%	38.00%	78.70%
Baseline + Cold Fusion						
+ s^{AM} in gate	Source	7.25%	13.88%	15.63%	30.71%	50.56%
+ Fine-Grained Gating	Source	6.14%	12.08%	14.79%	30.00%	47.82%
+ ReLU layer	Source	**5.82%**	**11.52%**	14.89%	30.15%	48.40%
+ Probability Projection	Source	**5.94%**	**11.87%**	**13.72%**	**27.50%**	**38.17%**

Table 2: *Speech recognition results for the various models discussed in the paper. The CTC model is based on Deep Speech 2 [14] architecture.*

internalized a decent language model of the LibriSpeech ASR corpus.

When the baseline decoder is augmented with a LibriSpeech neural language model via Deep Fusion, test results slightly improve in noisy settings as one would expect from the gating mechanism favoring language information at times of input uncertainty. But, Deep Fusion models perform worse on the clean test set than vanilla attention. Learning to first discount the inherent bias of the ASR attention decoder is challenging when both the attention decoder and LM decoder weights are fixed and a scalar gate is used to weigh the language model. Additional information isn't utilized effectively.

With Cold Fusion, results improve 10 to 20% relatively across all test sets including the out-of-domain read movie dialog data set (final column in Table 1). By allowing the attention decoder and a more expressive fusion layer to learn to interact with each other during training, Seq2Seq can make better use of the language model for the end task.

In addition to improved generalization performance, the Cold Fusion model also converges $3\times$ faster than the baseline Seq2Seq model to reach the same loss reducing training time considerably (see Figure 1).

4.3. Improved Domain Transfer

Swapping the language model is not possible with Deep Fusion because of the state discrepancy issue motivated in Section 3. All fusion models were therefore trained and evaluated with the same language model that achieved a low perplexity on both source and target domains. This way, we can measure improvements in transfer capability over Deep Fusion due to the training and architectural changes.

Table 2 compares the performance of Deep Fusion and Cold Fusion on the source and target held-out sets. Deep Fusion has a bigger effect on both domains than on LibriSpeech data because these acoustic models were trained on much less data. However, Cold Fusion continues to consistently outperform the baselines on both metrics on both domains. For the task of predicting in-domain, our best model gets a relative improvement of more than 21% over the baseline and a relative improvement of 15% over Deep Fusion.

We get even bigger improvements in out-of-domain results. The baseline attention model that was trained on the source domain gets a significantly worse WER on the target domain compared to a similar model trained directly on the target domain. The goal of domain adaptation is to bridge the gap between these numbers. The final column in Table 2 shows the remaining gap as a fraction of the difference for each model. As evident from the table, Deep Fusion decreases the domain gap to 76.57% while Cold Fusion reduces it down to 38.17%.

We also performed an ablation study to understand the effects of various architectural changes over the Deep Fusion layer. Leveraging the language model probability instead of the

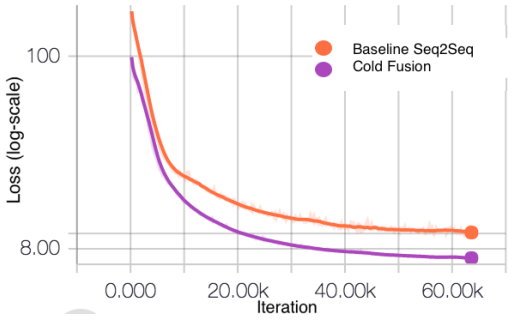

Figure 1: *Cross-entropy loss on the search queries dev set for the baseline model (orange) and the proposed model (purple) as a function of training iteration. Training with a language model speeds up convergence considerably.*

language model state in the fusion layer substantially helps with generalization, although this method is somewhat impractical for word-based models on large-vocabulary tasks. Intuitively, the character probability space shares the same structure across different domains unlike the hidden state space.

4.4. Decoder Efficiency

We test whether Cold Fusion does indeed relieve the decoder of learning an implicit language model. We do so by checking how a decrease in the decoder capacity affected the error rates. As evidenced in Table 3, the performance of the Cold Fusion models degrades gradually as the decoder cell size is decreased whereas the performance of the attention models deteriorates abruptly beyond a point. It is remarkable that the Cold Fusion decoder still outperforms the full attentional decoder with $14\times$ smaller decoder capacity.

Table 3: *Effect of decoder dimension on the model's performance. The performance of Cold Fusion models degrades more slowly as the decoder size decreases indicating that the effective task capacity is much larger with fusion.*

Model	Decoder size	Source	
		CER	WER
Attention	64	16.33%	33.98%
	128	11.14%	24.35%
	256	8.89%	18.74%
	960	7.54%	14.68%
Cold Fusion	64	9.47%	17.42%
	128	7.96%	15.15%
	256	6.71%	13.19%
	960	5.82%	11.52%

4.5. Fine-tuning for Domain Adaptation

In the presence of limited data from the target distribution, fine tuning a model for domain transfer is often a promising approach. We test how much labeled data from the target distribution is required for Cold Fusion models to effectively close the domain adaptation gap.

The same language model from Section 4.3 trained on both the source and target domains was used for all fine-tuning experiments. We fine-tuned only the fusion mechanism of the best Cold Fusion model from Table 2 on various amounts of the labeled target dataset.

Table 4: *Results for fine-tuning the acoustic model (final row from Table 2) on subsets of the target training data. *The final row represents an attention model that was trained on all of the target domain data.*

Model	Target Data	Target		
		CER	WER	Domain Gap
Cold Fusion	0%	13.72%	27.50%	38.17%
Cold Fusion + finetuning	0.6%	11.98%	23.13%	21.30%
	1.2%	11.62%	22.40%	18.49%
	2.4%	10.79%	21.05%	13.28%
	4.8%	10.46%	20.46%	11.00%
	9.5%	10.11%	19.68%	7.99%
Attention*	100%	8.84%	17.61%	0.00 %

Results are presented in Table 4. With just 0.6% of labeled data, the domain gap decreases from 38.2% to 21.3%. With less than 10% of the data, this gap is down to only 8%. Since only the fusion parameters were fine-tuned, all of the gains we see in these experiments come from combining the acoustic model and the language model in a manner more suitable to the target domain.

5. Conclusion

In this work, we presented a new general Seq2Seq model architecture where the decoder is trained together with a pre-trained language model. We study and identify architectural changes that are vital for the model to fully leverage information from the language model, and use this to generalize better; by leveraging the RNN language model, Cold Fusion reduces word error rates by up to 18% relative for clean and 20% relative for noisy speech compared to Deep Fusion on LibriSpeech. Additionally, we show that Cold Fusion models can transfer more easily to new domains, and with only 10% of labeled data nearly fully transfer to the new domain.

6. References

[1] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *ICLR*, 2015.

[2] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 4945–4949.

[3] W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals, "Listen, attend and spell," *arXiv preprint arXiv:1508.01211*, 2015.

[4] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.

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

[6] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proceedings of the 27th International Conference on Neural Information Processing Systems*, ser. NIPS'14. Cambridge, MA, USA: MIT Press, 2014, pp. 3104–3112. [Online]. Available: http://dl.acm.org/citation.cfm?id=2969033.2969173

[7] 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.

[8] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," 2014, http://arxiv.org/abs/1409.3215.

[9] Z. Yang, B. Dhingra, Y. Yuan, J. Hu, W. W. Cohen, and R. Salakhutdinov, "Words or characters? fine-grained gating for reading comprehension," *arXiv preprint arXiv:1611.01724*, 2016.

[10] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an ASR corpus based on public domain audio books," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015.

[11] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

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

[13] R. Collobert, C. Puhrsch, and G. Synnaeve, "Wav2letter: an end-to-end convnet-based speech recognition system," *arXiv preprint arXiv:1609.03193*, 2016.

[14] D. Amodei, R. Anubhai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, J. Chen, M. Chrzanowski, A. Coates, G. Diamos *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," *arXiv preprint arXiv:1512.02595*, 2015.

[15] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735—1780, 1997.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[17] 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.

[18] E. Battenberg, J. Chen, R. Child, A. Coates, Y. Gaur, Y. Li, H. Liu, S. Satheesh, D. Seetapun, A. Sriram, and Z. Zhu, "Exploring neural transducers for end-to-end speech recognition," *arXiv preprint arXiv:1707.07413*, 2017.

[19] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 1171–1179.