Cold Fusion: Training Seq2Seq Models together with Language Models

Philipp Meier

Department of Computational Linguistics, Heidelberg

Table of contents

Related Work

Motivation

- High quality parallel corpora is essential for good results in various NLP tasks
- Lack of such in low resource language pairs (Turkish-English) or tasks with domain restrictions (SMS chats)
- Integrate a language model in a sequence to sequence model to give 'hints' and improve fluency
- Multiple methods to integrate a language model
- Assumption: NMT model and RNN language model have been pre-trained separately before integration

Translation Model Architecture

- Model after Bahdanau et al., 2014
- Encoder consists of forward and backward RNNs
- Decoder: Single Layer RNN with soft alignment mechanism
- Decoders hidden states: $s_t^{TM} = f_r(s_{t-1}^{TM}, y_{t-1}, c_t)$
- Deep output layer to compute the conditional distribution: $p(y_t|y_{<t}, x) \propto exp(y_t^\top (W_\circ f_\circ(s_t^{TM}, y_{t-1}, c_t) + b_\circ))$
- Training objective: Maximize the conditional log-likelihood of the bilingual training corpus:

$$\max_\theta \frac{1}{N} \sum_{n=1}^{N} log p_\theta(y^{(n)}|x^{(n)})$$

Shallow fusion I

- Integration of recurrent neural network language model in the decoder
- Translation Model (TM) proposes set of candidate words at each time step t
- Candidates are scored according to the sum of the scores given by the Language Model and Translation Model.
- At each t, the model computes the score of every possible next word for each hypothesis of all hypotheses $\left\{ y^{(i)}_{\leq t-1} \right\}$
- Score = Score of the hypothesis + Score given by NMT to the next word
- Sort new hypotheses according to their scores, select top K as candidates $\left\{ \hat{y}^{(i)}_{\leq t} \right\}_{i=1,\ldots K}$

Shallow fusion II

- Rescore hypotheses with the weighted sum of the scores by the NMT and the LM
- Only recompute the score of the 'new word' at the end of each candidate hypothesis
- $\log p(y_t = k) = \log p_{TM}(y_t = k) + \beta \log p_{LM}(y_t = k)$

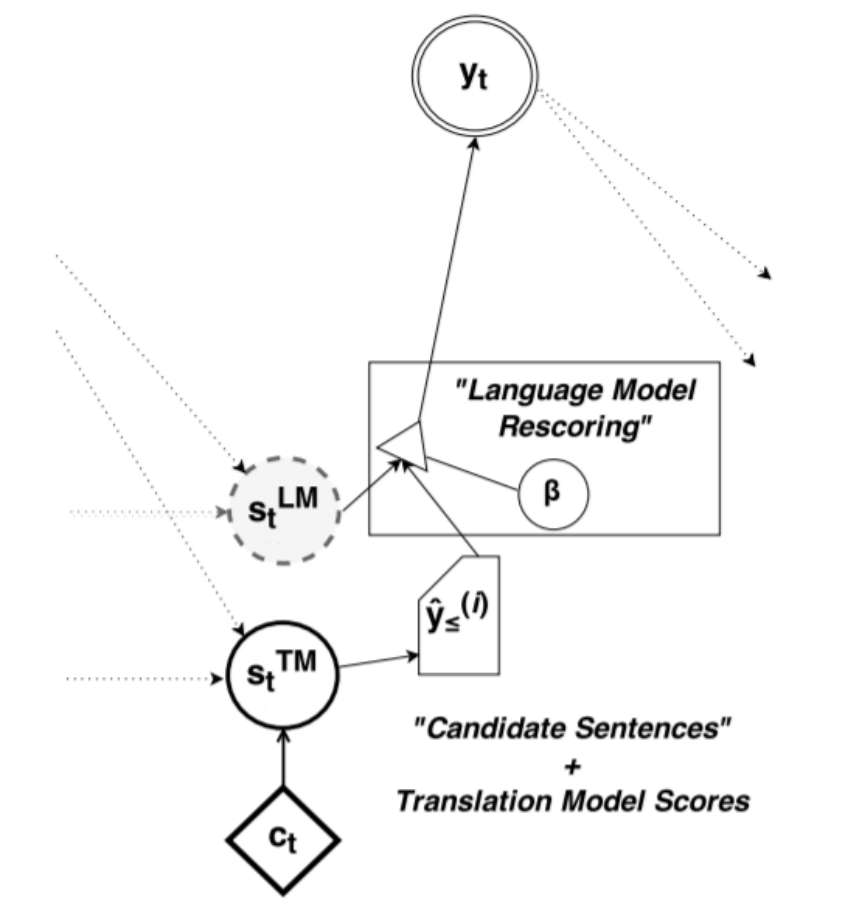


Figure 1: Shallow Fusion [5]

Deep fusion I

- Stronger connection between decoder of the NMT and language model
- Concatenation of their hidden states
- Finetuned to use both hidden states when computing the output probability of the next word
- $p(y_t|y_{<t}, x) \propto exp(y_t^\top (W_\circ f_\circ(s_t^{LM}, s_t^{TM}, y_{t-1}, c_t) + b_\circ))$

Deep fusion II

- Balancing of LM and TM via gating mechanism
- $g_t = \sigma\left(v_g^\top s_t^{LM} + b_g\right)$
- $s_t^{DF} = [c_t, s_t^{TM}; g_t s_t^{LM}]$
- $y_t = softmax(DNN(s_t^{DF}))$

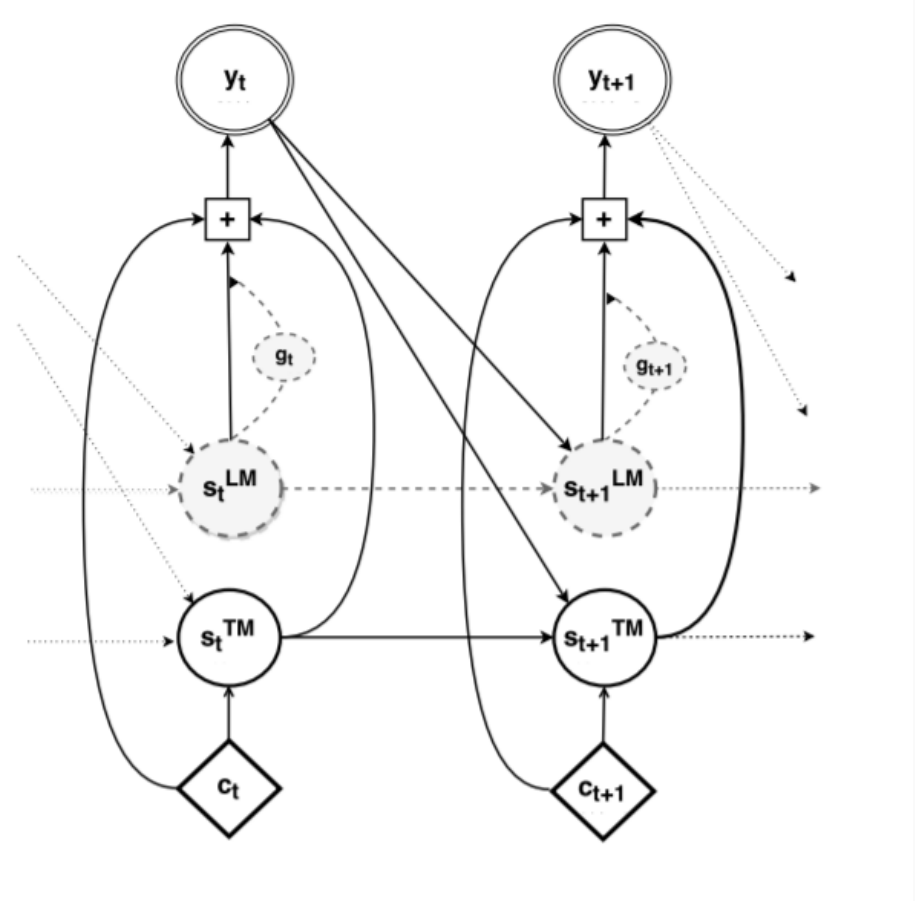


Figure 2: Deep Fusion [5]

Semi-supervised Learning in Seq2Seq Models

- Backtranslation
  - Combine monolingual training data with automatic backtranslation
  - Backtranslate monolingual target text into source language
  - Increase the parallel training corpus by translating the unlabeled target domain text
- Unsupervised pre-training
  - Weights of encoder and decoder are initialized with weights of two pre-trained language models
  - Finetuned with labeled data
  - Idea: Find a good initialization point

Motivation for Cold Fusion

- Deep Fusion has drawbacks
- Translation and Language model are trained separately, so the decoder of the TM learns from training data labels
- Waste of decoder capacity for redundant information
- Bias towards the training labels of the parallel corpus which limits deployability

Cold Fusion

Background

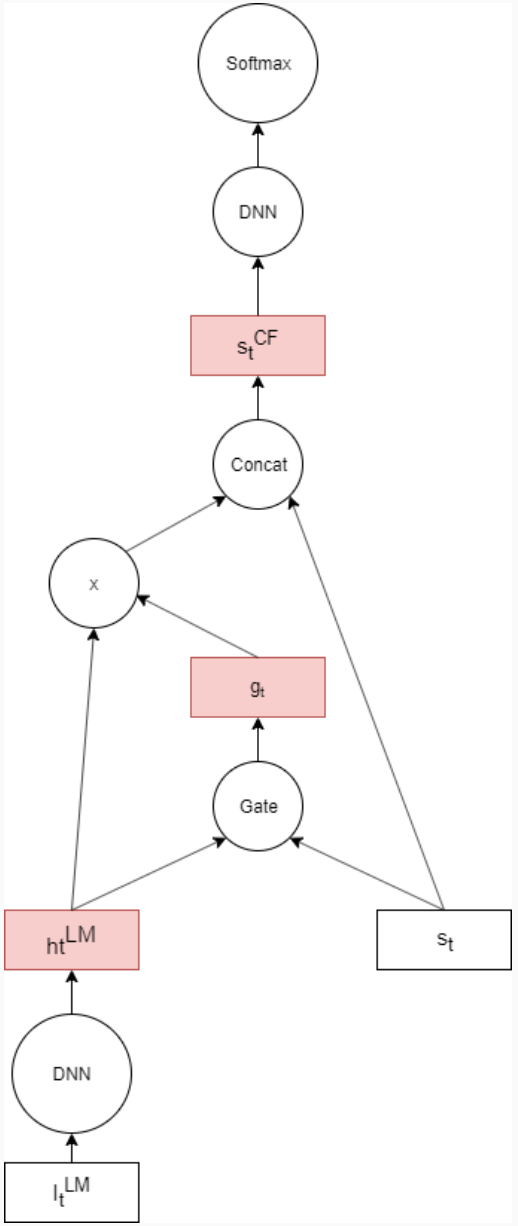
- Sequence-to-Sequence Models
  - Encoder maps input sequence $x = (x_1, ..., x_T)$ to representation h
  - Decoder which generates an output sequence $y = (y_1, ...y_K)$ using h
- Inference and Language Model Integration
  - Compute the most likely sequence during inference
  - $\hat{y} = \textbf{argmax}_y \, log \; p(y|x)$
  - Use of left-to-right beam search because argmax is intractable
  - Integration through change of the inference task to:
  - $\hat{y} = \textbf{argmax}_y \; log \; p(y|x) + \lambda \; log \; p_{LM}(y)$
  - Shallow Fusion uses this way of integration

Cold Fusion Properties

- Seq2Seq model is trained from scratch together with a pre-trained language model $\rightarrow$ Early Training Integration
- Cold Fusion uses a char based RNN as language model
- Both hidden states, s_t and s_t^{LM} can be used for the gate computation
- Use of fine-grained gating mechanism, different gate value for each hidden node
- Use of language models probability instead of hidden state for better generalization

Cold Fusion Layer

- $h_t^{LM} = DNN(\ell_t^{LM})$
- $g_t = \sigma(W[s_t; h_t^{LM}] + b)$
- $s_t^{CF} = [s_t; g_t \circ h_t^{LM}]$
- $r_t^{CF} = DNN(s_t^{CF})$
- $\hat{P}(y_t|x, y_{<t}) = softmax(r_t^{CF})$



Fine grained Gating

- In Yang et al [8]: Mechanism to combine word-level and character-level representation
- In Cold Fusion:
  - $g_t = \sigma(W_g[s_t^{ED}; s_t^{LM}] + b_g)$
  - $s_t^{CF} = [S_t^{ED}; g_t \circ s_t^{LM}]$
- In Deep Fusion:
  - $g_t = \sigma(v_g^T d_t^{LM} + b_g)$
  - $d_t^D F = [c_t; d_t; g_t d_t^{LM}]$
- Vector values are used instead of scalar values

Deep & Cold Fusion Architecture

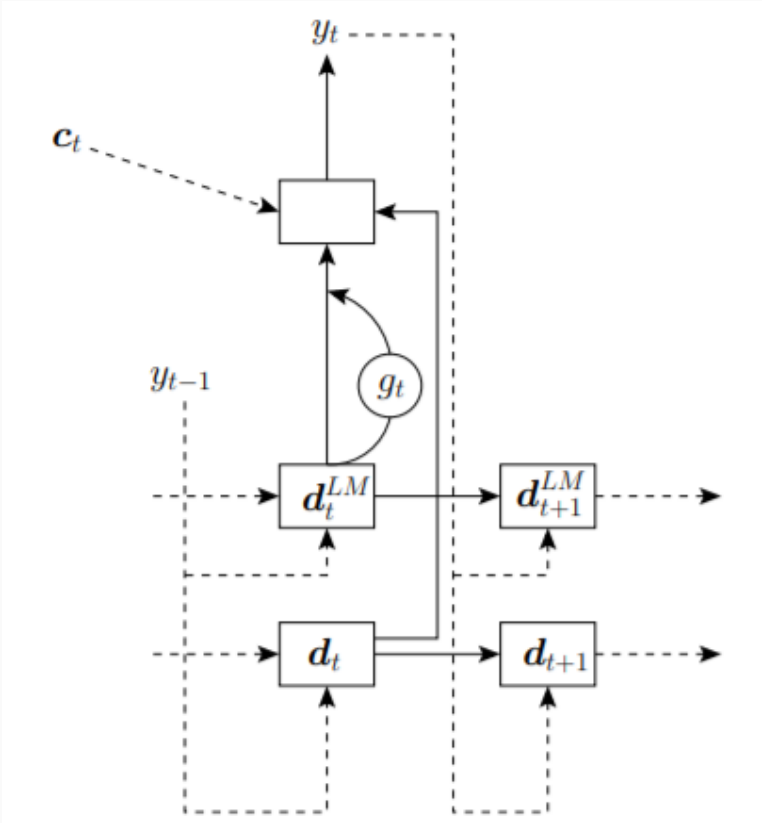


Figure 3: Deep Fusion

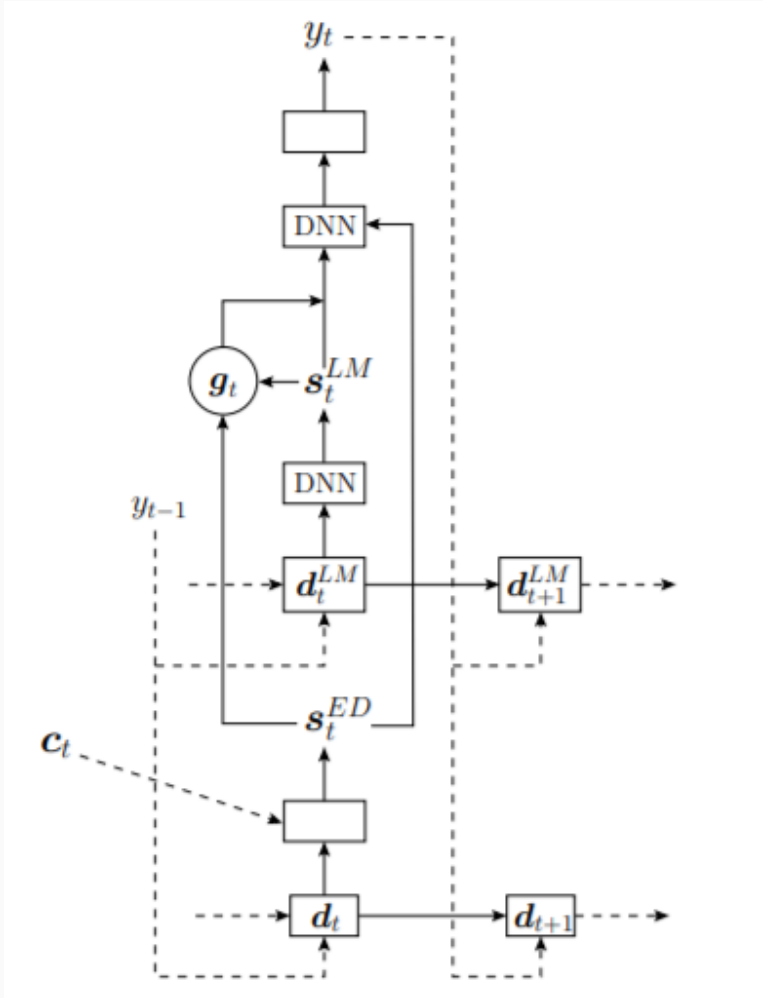


Figure 4: Cold Fusion

Experiments

Datasets

- Evaluation on the Speech Recognition Task
- Source: based on search queries, 411,000 utterances, 650 hours of audio
- Target: based on movie transcripts, 345,000 utterances, 676 hours of audio
- Used Amazon Mechanical Turk to provide audio recordings
- Held out of 2048 from each domain for evaluation

Architectures

- Language Model
  - Trained on 25 million words
  - Three layers of GRU, hidden state dimension of 1024
  - Minimize Cross Entropy of predicting the next character given the previous characters
  - Adam optimizer with a batch size of 512
- Acoustic Models
  - Seq2Seq model with soft attention
  - Encoder: 6 BLSTM with a dimension of 480
  - Decoder: Single Layer with 960 dimensional GRU with hybrid attention

Language Model Perplexity on Dev set

Model	Domain	Word Count	Perplexity	
			Source	Target
GRU (3*512)	Source	5.73M	2.670	4.463
GRU (3*512)	Target	5.46M	3.717	2.794
GRU (3*1024)	Full	25.16M	**2.491**	**2.325**

Table 1: Perplexities on the Development set [6]

Cross Entropy Loss

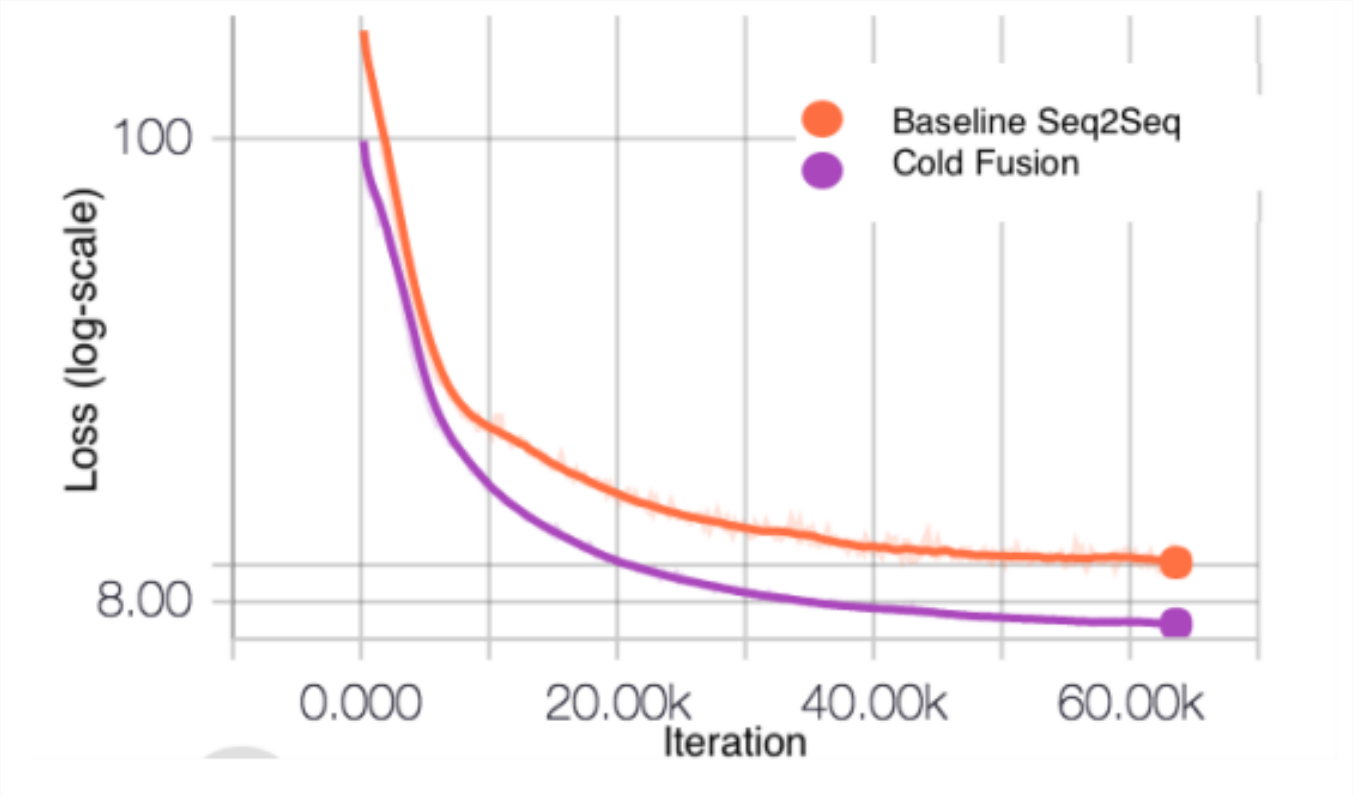


Figure 5: Cross-entropy loss on dev set, baseline (orange), proposed model (purple) [6]

Speech Recognition Results

Model	Train Domain	Test on Source		Test on Target		
		CER	WER	CER	WER	Domain Gap
BL Attention Model	Source	7.54%	14.68%	23.02%	43.52%	100%
BL Attention Model	Target			8.84%	17.61%	0%
Baseline + Deep Fusion	Source	7.64%	13.92%	22.14%	37.45%	76.57%
+ s^{AM}*ingate*	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}*ingate*	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	**5.64%**	**11.87%**	**13.72%**	**27.50%**	**38.17%**

Table 2: Speech recognition results [6]

Speech Recognition Results

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 3: Results from models trained on the publicly available Librispeech (reading task) data. Results from the Wav2Letter model [4] are presented for reference. MFCC, Power Spectrum and Raw Wave are names for different features

Decoder Size

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%

Table 4: Effect of decoder dimension [6]

Fine-tuning for Domain Adaption

Model	Target Data	Target CER	Target WER	Target 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%

Table 5: Fine tuning of the acoustic model [6]

Comparison

Comparison

Shallow Fusion	Deep Fusion	Cold Fusion
late LM integration Separated until score computation late training integration	early LM integration LM directly integrated late training integration	early LM integration LM directly integrated early training integration

Table 6: Comparison between fusion methods

Architectures & Data

LAS Model

- Encoder: 4-layer pyramidal bidirectional LSTM network, 256 hidden units in each direction of the layer
- Encoder for GVS: 5 unidirectional LSTM layers of 1400 hidden units each
- Decoder: Single Layer unidirectional LSTM with 256 hidden units
- Decoder for GVS: 2 unidirectional LSTM layers of 1024 hidden units each

Comparison II

Model	Switchboard	Call Home	Full
LAS	17.1	27.9	22.6
Shallow Fusion	**15.6**	**26.6**	**21.1**
Deep Fusion	16.3	27.2	21.7
Cold Fusion	16.3	27.3	21.8

Table 7: Word error rates (WER) on Eval2000 [7] for the baseline model and fusion approaches, SWB = switchboard, CH = CallHome, Full = Eval2000

Comparison III

Model	VS14K	D15K
LAS	5.6	4.0
Shallow Fusion	**5.3**	3.7
Deep Fusion	5.5	4.1
Cold Fusion	**5.3**	3.9

Table 8: WER (%) on Google voice search (VS14K) and dictation data sets (D15K) for the baseline and fusion approaches. VS14K: 14 000 voice search utterances, D15K: 15 000 dictation utterances [7]

Second Pass Rescoring

Model	VS14K(oracle)	D15K(oracle)
LAS	5.4(2.2)	3.9 (1.5)
Shallow Fusion	5.3 (2.4)	**3.7** (1.6)
Deep Fusion	5.4 (2.0)	4.0 (1.5)
Cold Fusion	**5.0** (1.78)	3.8 (1.2)

Table 9: Word error rates for rescoring in Google data sets [7], Second pass rescoring with a large, production-scale LM [2]

Findings

Deep Fusion does not scale well with data $\rightarrow$ No gain over the baseline for large scale Google sets

Shallow Fusion performs quite well

Cold Fusion has strengths on the oracle task

Questions

- What do the authors mean in 3.3: ”LM state is not invariant to the permutation of state hidden nodes”?
- What do the authors mean by: ”Since logits can have arbitrary offsets, the maximum value is subtracted off before feeding into the layer” (p.3, lower right)?
- Is it possible (or even necessary) to include multiple language models, for example for translational tasks?

Questions

What is 'Adam' and how does it work?

- Optimization Method, derived from 'adaptive moment estimation.'
- Computes a learning rate per parameter, instead of a strict learning rate like in SGD
- Compute decaying averages of past and past squared gradients m_t and v_t
- $m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$
- $v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$
- To counteract biases a bias-correct fist and second moment estimates are computed
- $\hat{m}_t = \frac{m_t}{1-\beta_1^t}$; $\hat{v}_t = \frac{m_t}{1-\beta_1^t}$
- Update rule: $w_t = w_{t-1} - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$ [1]
- $w = weights, \eta = stepsize$

Questions

- Why this architecture is called ”Cold Fusion”
  - Due to the early training integration of the Language Model.
- Why not integrate a language model on the encoder side as well?
- Also, the language models are different: The one used in the Cold Fusion paper seems to be purely character based, whereas the one in the Deep Fusion paper (Gulcehre et al.) is also character based but constructed from a word-based language model. I am wondering whether the purely character based one is able to adequately represent phrase-level structures such as grammar. Besides, are whitespaces even part of the ”raw” ASR output? So are the predictions listed in Table 1 the actual outputs or are they tokenised?

Questions

Lastly, one question raised towards the end of the last session was whether there is an equivalent to residual connections for RNN-like models that mitigates the problem of vanishing gradients. My first intention was that Bahdanau-Attention achieves something like this by making constant the ”distance” in terms of weights and nonlinearities from each input to each output. If this is not correct, I would of course receive thankfully being corrected.

Questions

- In 2.3, why do the authors state that during warmstarting "... training on the parallel corpus could end up effectively erasing the knowledge of the language models."?
- In 3.2, how are the "different gate value for each hidden node" realised?
- Are the changes in model architecture from deep fusion to cold fusion necessary to train the seq2seq model alongside with the Language model. Why can't we do that with the deep fusion architecture?

Question

- This is a rather general question: I don't see why you would use specific domains (such as legal or medical documents (example in Sec. 1), search queries or movie transcripts (evaluation, Sec. 4.1)) for training in the first place. If you are going to use different domains and/or try to adapt to new domains anyway, doesn't it make more sense to train on broad corpora directly, so that they contain as many domains as possible in one go?
- "Since logits can have arbitrary offsets, the maximum value is subtracted off before feeding into the layer." (Sec. 3) - What is the maximum value in this context, and why would it be subtracted off? Can we see this anywhere in Formula 4?

Questions

- In the third step, the language model's hidden state is replaced with the language model probability, does it mean that in case of RNN, each cell receives the time step input and the prediction of the previous time step instead of the hidden state of the previous time step?
- Deep Fusion was also intended to help in cases of low resource language pairs. Though Cold Fusion can be easily used to transfer a model to a different domain, what about its usefulness in such a case? Maybe I've managed to miss it, but only the smaller training time of the decoder seems to be mentioned and the little data used to fine-tune to a different domain.

Question

- The DNN layer of Cold Fusion is a single affine layer with ReLU activation and then softmax. Any idea/intuition why this works better than anything else?
  - Unfortunately, this point is not further mentioned in the paper
  - Maybe to explicitly lead the softmax function?

Questions

- Can you briefly explain the ’hybrid attention’ mechanism? I didn’t really understand how it works and what the difference to ’normal’ attention is.
  - Extension of the attention mechanism with speech recognition features
  - Combines content and location information to select the next candidate for decoding

Question

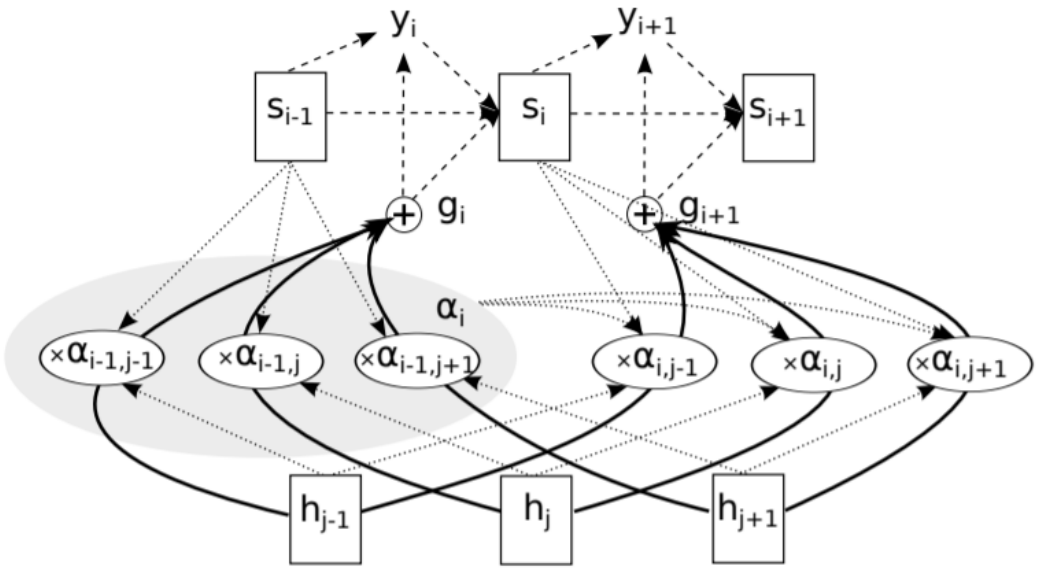


Figure 1: Two steps of the proposed attention-based recurrent sequence generator (ARSG) with a hybrid attention mechanism (computing α), based on both content (h) and location (previous α) information. The dotted lines correspond to Eq. (1), thick solid lines to Eq. (2) and dashed lines to Eqs. (3)–(4).

Figure 6: Hybrid Attention [3]

- 1) $\alpha = Attend(s_{i-1}, \alpha_{i-1}, h)$
- 2) $g_i = \sum_{j=1}^{L} {}_{i,j}, h_j$
- 3) $y \sim Generate(s_{i-1}, g_i)$
- 4) $s_i = Reccurency(s_{i-1}, g_i, y_i)$

Questions

- In section 3, in the third bullet point, they say they don't use the hidden states but the language model probability. And I assume they mean the logits (offset by the largest logit value?). Why not use the LM's softmax output? In my intuition, that would provide a probability representation that is more constant across language models.
- In section 4.1, they explain that they collect their own dataset. Why do they do that in the first place? Why not use an already existing dataset?

Questions

- How is the computation of the Domain Gap performed?
- When showing the impressive results for the domain Adaptation, the LM is trained on full domain. In the scenario were the LM trained only on the source domain, could we expect similar results?

References

[1] Vitaly Bushaev. Adam — latest trends in deep learning optimization. https://towardsdatascience.com/adam-latest-trends-in-deep-learning-optimization-6be9a291375c. Accessed: 2020-01-10.

[2] Chung-Cheng Chiu, Tara N Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjuli Kannan, Ron J Weiss, Kanishka Rao, Ekaterina Gonina, et al. State-of-the-art speech recognition with sequence-to-sequence models. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4774–4778. IEEE, 2018.

References ii

[3] Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. Attention-based models for speech recognition. In *Advances in neural information processing systems*, pages 577–585, 2015.

[4] Ronan Collobert, Christian Puhrsch, and Gabriel Synnaeve. Wav2letter: an end-to-end convnet-based speech recognition system. *arXiv preprint arXiv:1609.03193*, 2016.

[5] Caglar Gulcehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loic Barrault, Huei-Chi Lin, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. On using monolingual corpora in neural machine translation. *arXiv preprint arXiv:1503.03535*, 2015.

[6] Anuroop Sriram, Heewoo Jun, Sanjeev Satheesh, and Adam Coates. Cold fusion: Training seq2seq models together with language models. *arXiv preprint arXiv:1708.06426*, 2017.

References iii

[7] Shubham Toshniwal, Anjuli Kannan, Chung-Cheng Chiu, Yonghui Wu, Tara N Sainath, and Karen Livescu. A comparison of techniques for language model integration in encoder-decoder speech recognition. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 369–375. IEEE, 2018.

[8] Zhilin Yang, Bhuwan Dhingra, Ye Yuan, Junjie Hu, William W Cohen, and Ruslan Salakhutdinov. Words or characters? fine-grained gating for reading comprehension. *arXiv preprint arXiv:1611.01724*, 2016.