KNOWLEDGE DISTILLATION FOR RECURRENT NEURAL NETWORK LANGUAGE MODELING WITH TRUST REGULARIZATION

Yangyang Shi* *Mei-Yuh Hwang** *Xin Lei** *Haoyu Sheng**†

* Mobvoi AI Lab
† Williams College

ABSTRACT

Recurrent Neural Networks (RNNs) have dominated language modeling because of their superior performance over traditional N-gram based models. In many applications, a large Recurrent Neural Network language model (RNNLM) or an ensemble of several RNNLMs is used. These models have large memory footprints and require heavy computation. In this paper, we examine the effect of applying knowledge distillation in reducing the model size for RNNLMs. In addition, we propose a trust regularization method to improve the knowledge distillation training for RNNLMs. Using knowledge distillation with trust regularization, we reduce the parameter size to a third of that of the previously published best model while maintaining the state-of-the-art perplexity result on Penn Treebank data. In a speech recognition N-best rescoring task, we reduce the RNNLM model size to 18.5% of the baseline system, with no degradation in word error rate (WER) performance on Wall Street Journal data set.

Index Terms— Knowledge Distillation, LSTM, Language Model, Trust Regularization

1. INTRODUCTION

Recurrent Neural Networks are currently popular choices for language modeling. Recurrent Neural Network based language models (RNNLMs) have outperformed traditional N-gram based language models in many machine learning tasks, such as automatic speech recognition [1], machine translation [2] or text summarization [3].

To address the data sparsity issue in N-gram language models, RNNLMs represent each word in a continuous and lower dimensional space. In contrast to N-gram language models which only capture local short distance dependencies, RNNLMs are capable of representing full histories of variable lengths with recurrent vector representations.

However, in many natural language processing tasks, large models are usually required to achieve state-of-the-art performance. In deep model training, significant redundant representations are learned [4]. Due to their large memory footprint and daunting computational cost, the applications of large networks in practical scenarios are greatly hindered. To reduce the deep neural network model size, model quantization [5], model weight pruning [6], low rank matrix factorization [7] and knowledge distillation [8] are usually applied. In this paper, we consider the knowledge distillation method in the context of RNNLMs and their application in speech recognition N-best rescoring.

Knowledge distillation is also referred to as teacher student training, where a small model (student) is trained to match the output of a large model (teacher). In addition to model compression, knowledge distillation has been used in transfer learning and domain adaptation tasks. Depending on the specific scenario, the teacher and student models can be trained on the same or different data [8]. In this work, both of the teacher and student models are trained on the same data but with different model sizes.

In knowledge distillation, the student model is learned to minimize the combination of cross-entropy loss based on training data labels and Kullback-Leibler (KL) divergence to teacher model distribution. In this paper, our experiments in the context of language modeling show that the knowledge distillation methods [8] that use cross-entropy loss and KL divergence with fixed interpolation weights get worse results than using KL divergence alone. Hence we propose a trust regularization (TR) method to dynamically adjust the combination weights for these two types of losses. Two experiments are performed to verify the effectiveness of the proposed method. In the first experiment, the student model achieves state-of-the-art perplexity results on the Penn Treebank dataset [1] with a model size one third of that of the previously published best model. The second experiment is speech recognition N-best rescoring on Wall Street Journal dataset [9], where the student model size is only 18.5% of that from its teacher model and yet achieves similar word error rates.

2. KNOWLEDGE DISTILLATION FOR RNNLMs

2.1. Teacher Model: A high-rank RNNLM with regularizations

In this paper, knowledge distillation is built on top of a high-rank RNNLM [10] with several RNNLM specific regularizations and optimization methods [11].

The high-rank RNNLM uses a mixture of softmaxes (MoS) to make the softmax layer in RNNLM more expressive. Similar to conventional RNNLMs [12, 11], a sequence of hidden states is obtained after processing the input sequence X over a stack of recurrent layers. On top of the hidden states, the MoS represents the conditional distribution of current word x_t as weighted sum of different softmax layers.

To achieve optimal performance for RNNLMs, the following regularization techniques [11] are applied.

- Three different dropouts: DropConnect [13] is the dropout to the weight matrices within LSTM cells for hidden to hidden transitions. Variational dropout is applied to all inputs and outputs for LSTM cells. Embedding dropout [14] is equivalent to using variational dropout on embedding layers.
- Activation Regularization is applied to penalize large hidden layer activations and to penalize dramatic difference in activations across neighboring frames.

However, in contrast to cross-entropy training, to achieve the best performance for knowledge distillation, our experiments reveal that all these regularization methods need to be turned off for student model training.

2.2. Knowledge Distillation

The basic idea of knowledge distillation [8] is to train a smaller student model by providing additional information in the form of outputs from a larger teacher model. Usually the student model, denoted by θ, is trained to minimize the interpolation of a cross-entropy loss according to training data labels (hard labels) and KL divergence between the student model outputs and the outputs from a teacher model (soft labels):

$$L(\theta) = \alpha L_{CE}(\theta) + (1-\alpha) L_{KL}(\theta) \tag{1}$$

where $L_{CE}(\theta)$ is the cross-entropy loss. In the context of language modeling, the cross-entropy loss can be represented as

$$L_{CE}(\theta) = -\sum_{x} 1(y=x) \log P(x|c,\theta) \tag{2}$$

where y is the hard label and $1(y=x)$ is the indicator function. The KL divergence of the student output distribution $P(x|c,\theta)$ to the teacher output distribution $Q(x|c,\theta_{te})$ can be formulated as

$$L_{KL}(\theta) = -\sum_{x} Q(x|c,\theta_{te}) \log \frac{P(x|c,\theta)}{Q(x|c,\theta_{te})}. \tag{3}$$

where $Q(x|c,\theta_{te})$ is the teacher model output distribution for each word x. $Q(x|c,\theta_{te}) \log Q(x|c,\theta_{te})$ is constant for each x since the teacher model θ_{te} is fixed. Therefore the minimization of the above equation is equivalent to minimization of the following loss

$$L_{KL}(\theta) = -\sum_{x} Q(x|c,\theta_{te}) \log P(x|c,\theta). \tag{4}$$

In [8], the output distribution $Q(x|c,\theta_{te})$ is represented as a softmax probability with temperature $\tau \geq 1$. We have tried different temperatures $\tau \in [1,2,5,8,10]$ in our experiments. We find that $\tau = 1$ gives the best performance and is used in all experiments reported here.

2.3. Trust Regularization

In language modeling, each training data label is represented as a degenerated data distribution which gives all probability mass to one class. So the degenerated data distribution is localized and can be over-confident, comparing with the teacher's probability distribution learned over the whole training data. Different from previous observations using knowledge distillation for acoustic modeling [8] and image classification [15], our experiment results show that the student model learned by minimizing the interpolation of cross-entropy loss and KL divergence performs worse than the student model learned by minimizing the KL divergence alone.

Thus, we propose the following trust regularization (TR) method to dynamically adjust the weight (trust) for cross-entropy loss.

$$L(\theta) = R(y) L_{CE}(\theta) + L_{KL}(\theta) \tag{5}$$

where $R(y)$ is the trust regularizer that is formulated as follows:

$$R(y) = -\alpha \sum_{x} 1(y=x) \log(1 - Q(x|c,\theta_{te})) \tag{6}$$

where $\alpha > 0$ is a scalar value. The more the teacher model $Q(x|c,\theta_{te})$ agrees with the hard label y, the more we trust CE and therefore the more weight the CE loss will receive.

3. EXPERIMENTS

To evaluate the proposed TR in knowledge distillation for RNNLMs, we conduct experiments on two tasks. The first task is to measure the perplexity of language models. The widely used benchmark dataset Penn Treebank (PTB) [1] is used. PTB data has a predefined $10K$ close vocabulary set.

The second task is to measure the WER of RNNLMs in speech recognition N-best rescoring on Wall Street Journal [9] (WSJ). For acoustic training on the SI-284 training set, and language modeling training data preprocessing, we follow the recipe in the Kaldi speech recognition toolkit [25]. There are $37M$ words in the WSJ LM trainining data, with

model	#Param	Valid	Test
RNN-LDA+KN-5+cache [16]	9M	-	92.0
LSTM [12]	20M	86.2	82.7
Variational LSTM medium MC [14]	20M	-	78.6
Variational LSTM large MC [14]	66M	-	73.4
Char-CNN-LSTM [17]	19M	-	78.9
Pointer-Sentinel [18]	21M	72.4	70.9
LSTM + continuous cache pointer [19]†	-	-	72.1
Tied variational LSTM+augmented loss [20]	24M	75.7	73.2
Tied variational LSTM+augmented loss [20]	51M	71.1	68.5
Variational RHN [21]	23M	67.9	65.4
NAS Cell [22]	25M	-	64.0
NAS Cell [22]	54M	-	62.4
4-layer skip connection LSTM [23]	24M	60.9	58.3
AWD-LSTM finetune [24]	24M	60.0	57.3
AWD-LSTM-Mos w/o finetune [10]	22M	58.1	56.0
Ours w/o finetune	7M	57.8	55.6
AWD-LSTM-Mos finetune [10]	22M	56.5	54.4
Our teacher model (AWD-LSTM-Mos finetune $\times$ 5)	22M$\times$ 5	52.7	51.4
Ours finetune	7M	**55.9**	**54.0**
AWD-LSTM-Mos finetune dynamic eval [10]†	22M	48.33	47.69
Ours finetune dynamic eval†	7M	**48.17**	**47.60**

Table 1. Single model perplexity results on validation and test sets on PTB. † indicates using dynamic evaluation.

$200K$ of which are separated as the validation data. The vocabulary of the WSJ language model is capped at $40K$ by the above Kaldi recipe. Those training words not in the $40K$ are mapped to token UNK. Low-frequency words in the $40K$ vocabulary are mapped to a special token as RNN_UNK (essentially a "rare-word" class) for RNNLM training.

3.1. Experiments on PTB

In this task, the teacher model is an ensemble of five models that are initialized based on different random seeds ($[31, 37, 61, 71, 83]$). To train each component LM in the teacher model, we closely follow the regularization, optimization and hyper-parameter tuning techniques that are applied by [24, 10]. Each component model has 3 layers of LSTM with 960 neurons. The embedding size is set to 280. Before the softmax layer, the LSTM output is projected down to a bottleneck layer with size 620. In the mixture of softmax layer, 15 experts are used. Dropout rates of 0.4, 0.29 and 0.225 are used for the LSTM input, the LSTM output, and hidden-to-hidden transition in LSTM, respectively. For the other layers, the dropout rate is set to 0.4. The number of parameters for each component model is $22M$.

The student model is trained to minimize the combined loss with trust regularization (Eq. 6) with $\alpha = 0.1$. It has 3 layers of LSTM with hidden size 480. The dimensions of the embedding layer and the bottleneck layer are 200 and 300, respectively. The student model uses the same number of experts in the softmax layer as the teacher model. We do not apply dropout or activation regularization in training the student model.

Table 1 gives the language model perplexity results on PTB data. The single student model trained via the trust regularized knowledge distillation method outperformed all the baselines with or without fine-tuning and dynamic evaluation. Using only one third (7/22) of parameters compared with the previously published best model [10], our student model achieved better perplexity.

3.1.1. Ablation Analysis

To further verify the contribution of the trust regularized knowledge distillation method, we conduct ablation experiments on PTB dataset. All the ablation experiments exclude the usage of fine-tuning and dynamic evaluation to avoid distractive factors. Hence the first row of Table 2 is copied from the first "7M" model in Table 1. Each of the following experiments in the table only changes one factor, compared to the first row.

The second row turns off cross entropy loss completely. In the third row, a constant weight of 0.1 is used for CE loss instead of a dynamic weight assigned by TR. The weight for KL loss remains to be 1 as shown by Eq. 5. Comparing these two, we find that KL loss alone is better than a constant CE weight. This demonstrates the effectiveness of the dynamic weight from TR.

Furthermore, as indicated by the next two rows in Table 2, we find that, different from language model learning

model	Valid	Test
student model ($\tau = 1.1$)	57.8	55.6
-cross-entropy loss ($\alpha = 0$)	58.5	56.8
-trust regularization	59.4	57.4
+dropout	65.4	62.9
+activation regularization	58.0	55.9
-knowledge distillation	67.8	65.7

Table 2. Ablation analysis on PTB data.

using hard labels, knowledge distillation method renders better models without dropout or activation regularization .

Finally, without using knowledge distillation (i.e. no teacher), the small model is trained with only cross-entropy loss. In such situation, the conventional regularization and optimization techniques such as dropout are applied to get the best performance for cross entropy training. It shows that without knowledge distillation, the perplexity increases by 17.3% and 18.2% on validation and test data.

3.1.2. *Trust Regularization vs. Fixed Weight Interpolation*

Table 3 gives the comparison between TR method and fixed weight interpolation in knowledge distillation. Again, to avoid distraction, neither fine tuning nor dynamic evaluation is used here to acquire the results. The results show that in language modeling, the fixed weight interpolation of cross-entropy loss and KL divergence renders worse models than using KL divergence loss alone or TR method.

model	Valid	Test
student model	57.8	55.6
0.0CE+1.0KL	58.5	56.8
0.1CE+0.9KL	59.5	57.6
0.2CE+0.8KL	59.7	57.6
0.5CE+0.5KL	63.5	58.2

Table 3. Comparison of constant interpolation vs. trust regularized interpolation for combining CE loss and KL loss.

3.2. Experiments on WSJ

In this experiment, the teacher is an ensemble of two models (Comp 1 and Comp 2 in Table 4) that are initialized with random seed 17 and 31. Each component model has one 900 dimensional embedding layer, 3 layers of LSTM that each has 1150 hidden neurons and one bottleneck layer with 650 neurons before softmax layer. In the mixture of softmaxes, 7 experts are used. A dropout rate of 0.4 is used for variational dropout. ConnectDrop isn't applied here. The embedding layer dropout rate is 0.1. In the student model, there are also one layer of embedding, 3 layers of LSTM and one bottleneck layer. Each layer of the student model has 250 neurons. There are also 7 experts in the mixture of softmaxes. For student model training, no dropout is applied. The trust regularization weight is set to 0.01.

Model	#Param	Valid ppl	Dev93/BD	Eval92/BD
1st-pass	-	118	9.02/6.30	6.31/3.90
RNNLM	2012M	81.2	7.44/4.82	4.98/2.66
Teacher	65×2M	**45.4**	**6.18/3.80**	**4.36/2.37**
Comp 1	65 M	53.2	6.30/4.10	4.34/2.52
Comp 2	65 M	53.8	6.31/4.12	4.51/2.60
CE only	12 M	72.5	6.53/4.41	4.54/2.76
fixed interp.	12 M	55.3	6.34/4.13	4.43/2.61
TR interp.	12 M	54.1	6.30/4.13	4.36/2.57

Table 4. Perplexity (ppl) and word error rates on WSJ data. "1st-pass" is the top1 output from ngram first-pass decoding. 'BD' means the first pass decoding uses a **b**ig **d**ictionary for pronunciation lookup.

The experimental results on WSJ are listed in Table 4. A first-pass decoding is run by using the ngram LM of $40K$ vocabulary generated by Kaldi scripts, to generate 1000 best hypotheses per utterance for rescoring. While compiling the decoding graph, any word in the $40K$ vocabulary will be removed if its phonetic pronunciation cannot be found or derived. We generate two sets of 1000-best hypotheses, each with a different pronunciation dictionary. BD in Table 4 means the nbest is generated from a bigger dictionary.

The student learned from TR knowledge distillation method has marginally better performance than the model trained based on fixed interpolation weights (CE weight 0.01). Compared to each of the mixture component models for the teacher, the student model could get almost identical error rates, but with only $12/65 \approx 18.5\%$ of parameters. The table also includes the performance of a pure CE training on the small model, without teachers. The 72.5 perplexity shows the importance of having a teacher.

4. CONCLUSIONS

In this paper, we apply knowledge distillation for RNNLMs. The experiments on language modeling reveal that the loss function using the fixed weight interpolation of cross-entropy loss and KL divergence renders worse models than KL divergence alone. To leverage the training hard labels for knowledge distillation, we proposed a trust regularization method to dynamically adjust the weight for cross-entropy loss. The experiments on PTB dataset showed that the student model trained via TR got the state-of-the-art perplexity using only one third of parameters. On the WSJ speech recognition N-best rescoring task, our knowledge distillation method reduced the model size to 18.5% of the best single model, with similar word error rates.

5. REFERENCES

[1] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernocký, and Sanjeev Khudanpur, “Recurrent neural network based language model,” in *The Proceedings of Interspeech*, 2010, pp. 1045–1048.

[2] Holger Schwenk, Anthony Rousseau, and Mohammed Attik, “Large, Pruned or Continuous Space Language Models on a GPU for Statistical Machine Translation,” *NAACL-HLT workshop on the Future of Language Modeling for HLT*, pp. 11–19, 2012.

[3] Alexander M. Rush, Sumit Chopra, and Jason Weston, “A Neural Attention Model for Abstractive Sentence Summarization,” *CoRR*, vol. arXiv:1509.00685, 2015.

[4] Misha Denil, Babak Shakibi, Laurent Dinh, Marc’Aurelio Ranzato, and Nando de Freitas, “Predicting Parameters in Deep Learning,” *CoRR*, vol. arXiv:1306.0543, 2013.

[5] Darryl Lin, Sachin Talathi, and Sreekanth Annapureddy, “Fixed Point Quantization of Deep Convolutional Networks,” *International Conference on Machine Learning*, vol. 48, pp. 2849–2858, 2016.

[6] Abigail See, Minh-Thang Luong, and Christopher D. Manning, “Compression of Neural Machine Translation Models via Pruning,” *CoNLL-2016*, pp. 291–301, 2016.

[7] Jian Xue, Jinyu Li, and Yifan Gong, “Restructuring of deep neural network acoustic models with singular value decomposition,” *The proceedings of Interspeech*, 2013.

[8] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean, “Distilling the Knowledge in a Neural Network,” *CoRR*, vol. arXiv:1503.02531, 2015.

[9] Douglas B Paul and Janet M Baker, “The design for the wall street journal-based CSR corpus,” in *Proceedings of the workshop on Speech and Natural Language*, 1992, pp. 357–362.

[10] Zhilin Yang, Zihang Dai, Ruslan Salakhutdinov, and William W. Cohen, “Breaking the Softmax Bottleneck: A High-Rank RNN Language Model,” *CoRR*, vol. arXiv:1711.03953, 2017.

[11] Stephen Merity, Nitish Shirish Keskar, and Richard Socher, “Regularizing and Optimizing LSTM Language Models,” *CoRR*, vol. arXiv:1708.02182, 2017.

[12] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals, “Recurrent Neural Network Regularization,” *CoRR*, vol. arXiv:1409.2329, no. 2013, 2014.

[13] Li Wan, Matthew Zeiler, Sixin Zhang, Yann LeCun, and Rob Fergus, “Regularization of neural networks using dropconnect,” *Proceedings of Machine Learning Research*, pp. 109–111, 2013.

[14] Yarin Gal and Zoubin Ghahramani, “A Theoretically Grounded Application of Dropout in Recurrent Neural Networks,” *CoRR*, vol. arXiv:1512.05287, dec 2015.

[15] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio, “FitNets: Hints for Thin Deep Nets,” *CoRR*, vol. arXiv:1412.6550, 2014.

[16] Tomas Mikolov and Geoffrey Zweig, “Context dependent recurrent neural network language model,” in *IEEE Workshop on Spoken Language Technology*, 2012, pp. 234–239.

[17] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush, “Character-Aware Neural Language Models,” *CoRR*, vol. arXiv:1508.06615, aug 2015.

[18] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher, “Pointer Sentinel Mixture Models,” *CoRR*, vol. arXiv:1609.07843, sep 2016.

[19] Edouard Grave, Armand Joulin, and Nicolas Usunier, “Improving Neural Language Models with a Continuous Cache,” *CoRR*, vol. arXiv:1612.04426, dec 2016.

[20] Hakan Inan, Khashayar Khosravi, and Richard Socher, “Tying Word Vectors and Word Classifiers: A Loss Framework for Language Modeling,” *CoRR*, vol. arXiv:1611.01462, 2016.

[21] Julian Georg Zilly, Rupesh Kumar Srivastava, Koutnik Jan, and Jürgen Schmidhuber, “Recurrent Highway Networks,” *CoRR*, vol. arXiv:1607.03474, jul 2016.

[22] Barret Zoph and Quoc V. Le, “Neural Architecture Search with Reinforcement Learning,” *CoRR*, vol. arXiv:1611.01578, nov 2016.

[23] Gabor Melis, Chris Dyer, and Phil Blunsom, “On the State of the Art of Evaluation in Neural Language Models,” *CoRR*, vol. arXiv:1707.05589, jul 2017.

[24] Stephen Merity, Nitish Shirish Keskar, and Richard Socher, “Regularizing and Optimizing LSTM Language Models,” *CoRR*, vol. arXiv:1708.02182, aug 2017.

[25] Daniel. Povey, Arnab. Ghoshal, Gilles. Boulianne, Lukas. Burget, Ondrej. Glembek, Nagendra. Goel, Mirko. Hannemann, Petr. Motlicek, Yanmin. Qian, Petr. Schwarz, Jan. Silovsky, Georg. Stemmer, and Karel. Vesely, “The Kaldi speech recognition toolkit,” *The Proceedings of ASRU*, pp. 1–4, 2011.