# PREDICTION OF LSTM-RNN FULL CONTEXT STATES AS A SUBTASK FOR N-GRAM FEEDFORWARD LANGUAGE MODELS

*Kazuki Irie, Zhihong Lei, Ralf Schlüter, Hermann Ney*

Human Language Technology and Pattern Recognition,
Computer Science Department, RWTH Aachen University, D-52056 Aachen, Germany
{irie, schlueter, ney}@cs.rwth-aachen.de, zhihong.lei@rwth-aachen.de

## ABSTRACT

Long short-term memory (LSTM) recurrent neural network language models compress the full context of variable lengths into a fixed size vector. In this work, we investigate the task of predicting the LSTM hidden representation of the full context from a truncated n-gram context as a subtask for training an n-gram feedforward language model. Since this approach is a form of knowledge distillation, we compare two methods. First, we investigate the standard transfer based on the Kullback-Leibler divergence of the output distribution of the feedforward model from that of the LSTM. Second, we minimize the mean squared error between the hidden state of the LSTM and that of the n-gram feedforward model. We carry out experiments on different subsets of the Switchboard speech recognition dataset for feedforward models with a short (5-gram) and a medium (10-gram) context length. We show that we get improvements in perplexity and word error rate of up to 8% and 4% relative for the medium model, while the improvements are only marginal for the short model.

***Index Terms***— language modeling, neural networks, knowledge distillation, student-teacher, speech recognition

## 1. INTRODUCTION

The main approach for achieving state of the art results in language modeling is the recurrent neural network (RNN) [1]. The RNN compresses the variable length context into a fixed size vector computed from the parameters shared over time. On the other hand, the n-gram feedforward language model (LM) is also interesting. When n is small, it has a potential to be directly integrated into the traditional decoding in automatic speech recognition (ASR) [2, 3]. However, in practice, a very long n-gram context (over 20 words) is needed for a feedforward LM to be competitive [4, 5] with the state of the art RNN using long short-term memory (LSTM) [6, 7] cells. The model based on the n-gram context does not know that the input it sees is only a truncated portion of the full context. We can consider training the model such that it has a chance to recover the truncated part of the context. If a well trained LSTM LM is available, it can compress the full context into a vector which can be paired to its truncated n-gram context. Learning such pairs is a vector to vector mapping problem suitable for a neural network. We explore it as a subtask to train n-gram feedforward LM. Since such an approach is a form of knowledge distillation [8, 9], we investigate the following two methods. The first approach is the standard transfer based on the Kullback-Leibler divergence of the output distribution of the feedforward model from that of the LSTM. Alternatively, we can minimize the mean squared error between the hidden state of the LSTM and that of the n-gram feedforward model. We carry out experiments on different subsets of the Switchboard corpus and demonstrate that the performance of the LSTM LM can be carried over to a medium context length n-gram feedforward model.

## 2. RELATED WORK

The model compression in Bucă et al.'s work [10] consists in labelling the unlabelled data by using a large ensemble of neural networks to generate data to train a single model. Such an idea of transferring the power of a large model or an ensemble into a single model has been extended by the use of the *soft label* in the works by Ba and Caruana [8] as *student teacher learning* and by Hinton et al. [9] as *knowledge distillation*. The technique has been used in multiple contexts of acoustic modeling [11, 12, 13, 14, 15]. In [16, 17], the transfer from an RNN was successfully used to improve feedforward acoustic models. The application in language processing tasks include the machine translation [18] and parsing [19]. As opposed to the aforementioned approaches in which the student is a neural network, in the early work by Deoras et al. [20, 21] a student n-gram count LM is trained by using the text data sampled from a teacher RNN LM. We investigate the possibility to transfer the LSTM performance into a short (5-gram) and medium (10-gram) context length feedforward models.

## 3. KNOWLEDGE DISTILLATION IN LANGUAGE MODELING

### 3.1. Neural Language Model Topologies

We consider two model topologies: n-gram feedforward neural network as the *student* and LSTM-RNN as the *teacher*. The feedforward models are based on the fully connected multilayer perceptron (MLP) except in Sec. 4.6 where the

convolutional neural network (CNN) is used. All neural language models we consider use an output layer factorized using word classes [22]. This factorization has a direct consequence for the knowledge distillation (KD) framework as shown in the next section. We consider two context sizes for the student model: 5-gram and 10-gram. The focus of this work is to evaluate the potential of knowledge transfer to improve neural language models with a short (5-gram) and a medium (10-gram) context length. The baseline language model with a set of parameters $\theta$ and vocabulary $V$ is trained on a text of $T$ events $w_t$ with context $h_t$ by minimizing the cross entropy between the model's output and the ground truth:

$$L(\theta) = -\sum_{t=1}^{T}\sum_{w\in V}\delta_{w,w_t}\log(p_\theta(w|h_t)) = -\sum_{t=1}^{T}\log(p_\theta(w_t|h_t))$$

which is equivalent to the minimization of the perplexity. In the following, $p_\theta$ and $p_{\text{LSTM}}$ respectively denote the distributions of the student model to be trained and the teacher LSTM model which is assumed to be already trained.

### 3.2. Cross Entropy Knowledge Distillation (CE-KD)

A standard approach for knowledge distillation is to minimize the Kullback-Leibler divergence of the student model's output distribution from that of the teacher model:

$$KL(p_{\text{LSTM}}|p_\theta) = \sum_{t=1}^{T}\sum_{w\in V} p_{\text{LSTM}}(w|h_t)\log(\frac{p_{\text{LSTM}}(w|h_t)}{p_\theta(w|h_t)})$$

which is equivalent to minimizing the cross entropy:

$$\tilde{L}_{\text{CE-KD}}(\theta) = -\sum_{t=1}^{T}\sum_{w\in V} p_{\text{LSTM}}(w|h_t)\log(p_\theta(w|h_t)) \quad (1)$$

Since all neural language models we consider in this work use the class based factorization:

$$p_\theta(w_t|h_t) = p_\theta(w_t|h_t, g(w_t))\cdot p_\theta(g(w_t)|h_t) \quad (2)$$

where $g(.)$ defines the function which maps a word $w$ to its word class $g(w)$, Eq. (1) can be specifically adapted for the neural LMs with the class factorized outputs. Instead of directly substituting the class factorization of Eq. (2) into both $p_\theta$ and $p_{\text{LSTM}}$ in Eq. (1), we opt for minimizing the cross entropy on the word part and class part distributions separately, which gives the following objective function:

$$L_{\text{CE-KD}}(\theta) = -\sum_{t=1}^{T}\left(\sum_{c\in C} p_{\text{LSTM}}(c|h_t)\log(p_\theta(c|h_t)) + \sum_{u\in g(w_t)} p_{\text{LSTM}}(u|h_t, g(w_t))\log(p_\theta(u|h_t, g(w_t)))\right) \quad (3)$$

where $C$ denotes the set of word classes.

As reported in previous works, we also found that combining objective functions based on the hard and soft targets gives better results. The final objective function is therefore an interpolation as follows:

$$L_{\text{int-CE-KD}}(\theta) = \lambda L_{\text{CE-KD}}(\theta) + (1-\lambda)L(\theta) \quad (4)$$

where the optimal value for the interpolation weight $\lambda$ can be tuned to optimize the validation perplexity.

### 3.3. Distillation based on the mean squared error between hidden states (MSE-KD)

We consider the task of mapping an n-gram context to the full context LSTM-RNN hidden state using a feedforward neural network. Our objective is to use such a task as a subtask to improve n-gram feedforward language models. For that, we use an objective function based on the mean squared error between the state of the final LSTM layer $y_t^{(\text{LSTM})}$ in the teacher model and the state of the final hidden layer in the feedforward student LM $y_t(\theta)$:

$$L_{\text{MSE}}(\theta) = \frac{1}{T}\sum_{t=1}^{T}||y_t^{(\text{LSTM})} - y_t(\theta)||_2^2 \quad (5)$$

Like any multitask learning approach, we scale and add this objective function to the original objective function:

$$L_{\text{int-MSE-KD}}(\theta) = \lambda L_{\text{MSE}}(\theta) + L(\theta) \quad (6)$$

It can be noted that as opposed to the CE-KD case, this approach does not require computation of the output distribution of the teacher model. Therefore, MSE-CE has a practical advantage over the CE-KD when a large vocabulary is used and class based factorization is not used. The computation of MSE in Eq. (5) requires the teacher and the student to have the same dimension at the penultimate layer. To simplify the comparison between CE and MSE approaches, we tie these dimensions in all cases. In addition, we can initialize the parameters of the output layer in the student model by that of the teacher model since the dimensions match in this condition[1].

## 4. TEXT-BASED EXPERIMENTS

### 4.1. Data Description

We carry out experiments on different subsets of the Switchboard speech recognition dataset: the statistics are shown in Table 1. The cross validation (CV) set was prepared by randomly choosing sentences from the original Switchboard (3M) and Fisher (24M) transcriptions, resulting in 133K words (counting sentence end tokens). The rest of the transcriptions, which amounts to 26.7 M running words, are used as training data for all language models: both the 4-gram Kneser Ney count model (KN4) [23] and neural models. This

[1]Experimentally, we found that such an initialization only improves the perplexity by about 1% relative.

selection is the same as in [24]. A vocabulary size of 30K is used. The cross validation set was used for newbob tuning of the learning rate during neural LM training and for selecting the interpolation weight for combining the count LMs trained on the Switchboard and Fisher parts of the data. The Hub5_00 set is used as the development data to tune the LM scale for the recognition experiments in Sec. 5. We evaluate our models on the Hub5e_01 set.

**Table 1**. *Number of running words, OOV rates and average sentence lengths in word (Avg. len.) of all data sets and subsets used. The vocabulary size is 30K.*

| | | # Words | OOV[%] | Avg. len. |
|---|---|---|---|---|
| Train | | 26.7M | 1.6 | 11.2 |
| Cross Validation | | 133K | 0 | 12.8 |
| Hub5_00 | Total | 45K | 1.1 | 10.4 |
| | CH | 23K | 1.6 | 9.1 |
| | SWB | 22K | 0.7 | 12.3 |
| Hub5e_01 | | 65K | 1.0 | 11.4 |

### 4.2. Baseline Neural Language Model Setups

The teacher LSTM-RNN language model is composed of one projection layer of 600 nodes, one LSTM layer of 600 nodes, and the output layer. The output layer is factorized using 200 word classes trained using the exchange algorithm with the bigram two-sided criterion [25]. We use this LSTM model as the teacher model for all experiments. The student feedforward language models has one projection layer with 100 nodes for each word, two non-linear layers, and the output layer. The dimension of the final hidden layer is set to 600 since it is tied with that of the teacher (as discussed in Sec.3.3). The dimensions of the other hidden layers are optimized between 600, 1000, 1200 and 1500 for baseline models as well as when knowledge distillation is used (depending on cases, either 1000 or 1200 were found to work the best). We use the gated linear unit (GLU) activation function [5] [2] which transforms the input vector $x_t$ to the output vector $y_t^{(\mathrm{GLU})}$ by using the weight matrices $A$, $B$ and bias vectors $c$, $d$ as:

$$y_t^{(\mathrm{GLU})} = (Ax_t + c) \odot \sigma(Bx_t + d) \quad (7)$$

All neural networks are trained with the stochastic gradient using newbob learning rate scheduling. Batch size of 64 and 8 are respectively used to train the feedforward models and the LSTM model. We construct training sequences by concatenating sentences until that we get a sequence with more than 100 words. All neural language models are implemented using our toolkit rwthlm [26].

[2] We also found that the GLU converges faster than the sigmoid as reported in [5]. In our experiments, we observed that a sigmoid model can also reach the same level of perplexity, but it requires more epochs.

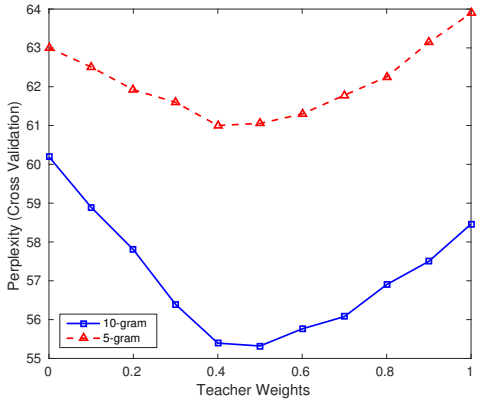


**Fig. 1**. *Effect of the teacher weight in the CE-KD case (Eq. 4) on the cross validation set.*

### 4.3. Results for CE-KD

We searched for the optimal value of the interpolation weight in Eq. (4) between 0 and 1: the cross validation perplexity results in Fig. 1 show that the optimal weights were 0.5 and 0.4 respectively for the 10-gram and the 5-gram. It should be noted that the pure knowledge distillation case $\lambda$=1 is better than the baseline case $\lambda$=0 for the 10-gram, while such is not the case for the 5-gram. Table 2 shows the perplexity results. We compute perplexities on the development set (Hub5_00) and the evaluation set (Hub5e_01) without using any context across the sentence boundaries such that they are consistent with the speech recognition setup. For the CV set, we report the perplexities using the context across sentence boundaries by concatenating multiple sentences as is the case during training as described in Sec.4.2. We observe consistent improvements by CE-KD for both the 10-gram and 5-gram cases. We note that, given the short average sentence lengths in the Switchboard data, as shown in Table 1, the baseline perplexities are close between the 5-gram and the 10-gram. Larger improvements by knowledge distillation can be observed when we consider longer sequences: Only in Table 3, we report perplexities computed by using contexts across sentence boundaries on Hub5_00 and Hub5e_01. We use the same sentence concatenation as for training (Sec.4.2). We observe up to 8% relative improvements for the medium case. Such improvements are also potentially interesting for speech recognition, which we will investigate in the future work.

**Table 2**. *Perplexity results of **cross entropy knowledge distillation** (CE-KD).*

| LM | CE-KD | CV | Hub5_00 | Hub5e_01 | #Param. |
|---|---|---|---|---|---|
| KN4 | - | 75.9 | 74.6 | 65.3 | 7M |
| LSTM | - | 52.2 | 60.8 | 52.4 | 39M |
| 5-g FF | | 64.1 | 64.9 | 57.0 | 24M |
| | × | **61.0** | **62.4** | **54.9** | |
| 10-g FF | | 60.9 | 64.2 | 55.4 | 25M |
| | × | **55.3** | **59.0** | **51.4** | |

**Table 3**. *Perplexity results of cross entropy knowledge distillation (CE-KD) using* ***contexts across sentence boundaries***.

| LM | CE-KD | Hub5_00 | Hub5e_01 |
|---|---|---|---|
| LSTM | - | 52.2 | 46.0 |
| 5-g FF | | 62.2 | 54.1 |
| | × | **59.5** | **51.9** |
| 10-g FF | | 60.3 | 51.8 |
| | × | **54.7** | **47.6** |

### 4.4. Results for MSE-KD

The MSE objective function aims to fit the the GLU output (Eq. 7) in the student model to the LSTM state. Therefore, we can rather use the gated tangent unit (GTU) for the final hidden layer[3] in the student model:

$$y_t^{(\mathrm{GTU})} = \tanh(Ax_t + c) \odot \sigma(Bx_t + d) \quad (8)$$

which is used in the LSTM. After search in {0.2, 0.15, 0.1, 0.05, 0.01, 0.001}, we found 0.01 and 0.05 to be optimal for the 5-gram and 10-gram cases respectively. Table 4 shows that the GTU effectively gives slightly better perplexities than the GLU. Though the MSE-KD improves both 5-gram and 10-gram baseline models, CE-KD (Table 2) gives better PPLs.

**Table 4**. *Perplexity results for MSE-KD using the gated linear unit (GLU) or the gated tangent unit (GTU) in the final hidden layer. The baseline perplexities are copied from Table 2 for easy comparison.*

| Context | MSE-KD | Type | CV | Hub5_00 | Hub5e_01 |
|---|---|---|---|---|---|
| 5-gram | | - | 64.1 | 64.9 | 57.0 |
| | × | GLU | 63.0 | 64.3 | 56.4 |
| | × | GTU | **61.4** | **63.4** | **55.4** |
| 10-gram | | - | 60.9 | 64.2 | 55.4 |
| | × | GLU | 57.9 | 61.8 | 53.7 |
| | × | GTU | **56.4** | **60.5** | **52.6** |

### 4.5. MLP vs CNN as the student model

We evaluate a 5-gram student CNN feedforward model. We use the CNN composed of 4 convolutional layers (200 filters for each layer, with the filter size 2 and the word dimension of 100) followed by one fully connected layer of dimension 600. The GLU activation is used in all layers. We only evaluate CE-KD, which gave better PPL than MSE-KD in the previous section. The results are shown in Table 5. We first observed that the baseline CNN gives slightly better baseline PPL than the MLP case. However, the gap disappears after distillation.

**Table 5**. *MLP vs. CNN in CE based distillation. The best PPLs for the MLP are copied from Table 2 for easy comparison. All modes are* ***5-grams***.

| Model | CE-KD | CV | Hub5_00 | Hub5e_01 | #Params. |
|---|---|---|---|---|---|
| MLP | | 64.1 | 64.9 | 57.0 | 24M |
| | × | **61.0** | **62.4** | **54.9** | |
| CNN | | 62.4 | 64.1 | 55.9 | 22M |
| | × | **61.1** | **62.6** | **55.0** | |

[3]For all other layers, we use the GLU which we found to achieve slightly better CV PPL than the GTU in our preliminary experiments.

## 5. SPEECH RECOGNITION EXPERIMENTS

### 5.1. ASR and Lattice Rescoring Setups

Our baseline ASR setup is based on the system presented in [24]. From that system, we only use one acoustic model based on a 5-layer bidirectional LSTM-RNN with 500 nodes in each layer and the rescoring pipeline is simplified by applying the lattice rescoring only once using a single neural LM. The lattice rescoring with all neural models is done using the push forward algorithm [27]. In addition to a beam pruning, we use a recombination pruning with order 9. The LM scores of the KN4 and the neural models are interpolated using the weight optimized on the CV set.

### 5.2. Results

Table 6 shows the word error rate (WER) results. For the 10-gram case, we observed significant improvements in WER from both CE and MSE based distillation on all subsets. Improvements in WER of up to 4% relative are obtained and the performance is competitive to the LSTM. In contrast, for the 5-gram case, the benefit from knowledge distillation does not seem to carry over to the ASR results.

**Table 6**. *WER results. All results are reported after* ***interpolation*** *with the baseline count model (KN4).*

| Model | KD Type | Hub5_00 (Dev) | | | | Hub5e_01 (Eval) | |
|---|---|---|---|---|---|---|---|
| | | CH | | SWB | | | |
| | | PPL | WER | PPL | WER | PPL | WER |
| KN4 | - | 80.5 | 19.2 | 68.8 | 10.5 | 65.3 | 15.0 |
| LSTM | - | 63.1 | 17.5 | 52.4 | **9.2** | 49.7 | **13.3** |
| 5g FF | | 65.8 | 17.8 | 56.8 | 9.6 | 53.8 | 13.9 |
| | CE | 64.7 | 17.8 | 55.8 | 9.5 | 52.9 | 13.7 |
| | MSE | 65.2 | 17.6 | 56.2 | 9.5 | 53.1 | 13.8 |
| 10g FF | | 65.0 | 17.7 | 55.0 | 9.5 | 52.0 | 13.8 |
| | CE | 62.0 | **17.4** | 52.3 | **9.2** | 49.7 | 13.4 |
| | MSE | 63.1 | 17.5 | 52.7 | **9.2** | 50.2 | **13.3** |

## 6. CONCLUSION AND FUTURE WORK

Experiments were carried out on Switchboard datasets. We observed that using a student-teacher approach, ASR performance using LSTM LMs can be carried over to feedforward LMs while it requires a large enough n-gram context. Further work will concentrate on the application of this approach for tasks with longer sentences, where the ASR performance gap between the feedforward model and the LSTM can be larger.

## 7. ACKNOWLEDGEMENTS

This work has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 694537, project "SEQCLAS"). The work reflects only the authors' views and the ERC Executive Agency is not responsible for any use that may be made of the information it contains. This research has also received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No. 644283, project "LISTEN". We thank Pavel Golik for suggestions for the experiments and Zoltán Tüske for sharing his Switchboard ASR setups.

## 8. REFERENCES


[1] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur, "Recurrent neural network based language model." in *Proc. Interspeech*, Makuhari, Japan, 2010, pp. 1045–1048.

[2] H. Schwenk and J.-L. Gauvain, "Connectionist language modeling for large vocabulary continuous speech recognition," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 1, Orlando, FL, USA, 2002, pp. 762–765.

[3] Y. Huang, A. Sethy, and B. Ramabhadran, "Fast neural network language model lookups at n-gram speeds," in *Proc. Interspeech*, 2017, pp. 274–278.

[4] Z. Tüske, K. Irie, R. Schlüter, and H. Ney, "Investigation on log-linear interpolation of multi-domain neural network language model," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Shanghai, China, Mar. 2016.

[5] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," in *Proc. Int. Conf. on Machine Learning (ICML)*, Sydney, Australia, Aug. 2017.

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

[7] M. Sundermeyer, R. Schlüter, and H. Ney, "LSTM neural networks for language modeling." in *Proc. Interspeech*, Portland, OR, USA, Sep. 2012, pp. 194–197.

[8] J. Ba and R. Caruana, "Do deep nets really need to be deep?" in *Advances in Neural Information Processing Systems (NIPS)*, vol. 27, Quebec, Canada, Dec. 2014, pp. 2654–2662.

[9] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *NIPS Deep Learning and Representation Learning Workshop*, Montreal, Canada, Dec. 2014.

[10] C. Buciluǎ, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *Proc. ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, Philadelphia, PA, USA, Aug. 2006, pp. 535–541.

[11] J. Li, R. Zhao, J.-T. Huang, and Y. Gong, "Learning small-size DNN with output-distribution-based criteria," in *Proc. Interspeech*, Singapore, Sep. 2014, pp. 1910–1914.

[12] J. Cui, B. Kingsbury, B. Ramabhadran, G. Saon, T. Sercu, K. Audhkhasi, A. Sethy, M. Nussbaum-Thom, and A. Rosenberg, "Knowledge distillation across ensembles of multilingual models for low-resource languages," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, New Orleans, LA, USA, Mar. 2017, pp. 4825–4829.

[13] L. Lu, M. Guo, and S. Renals, "Knowledge distillation for small-footprint highway networks," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, New Orleans, LA, USA, Mar. 2017, pp. 4820–4824.

[14] S. Watanabe, T. Hori, J. Le Roux, and J. R. Hershey, "Student-teacher network learning with enhanced features," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, New Orleans, LA, USA, Mar. 2017, pp. 5275–5278.

[15] J. H. Wong and M. J. Gales, "Sequence student-teacher training of deep neural networks," in *Proc. Interspeech*, San Francisco, CA, USA, Sep. 2016, pp. 2761–2765.

[16] W. Chan, N. R. Ke, and I. Lane, "Transferring knowledge from a RNN to a DNN," in *Proc. Interspeech*, Dresden, Germany, Sep. 2015, pp. 3264–3268.

[17] K. J. Geras, A.-R. Mohamed, R. Caruana, G. Urban, S. Wang, O. Aslan, M. Philipose, M. Richardson, and C. Sutton, "Blending LSTMs into CNNs," *in Workshop Track of International Conference on Learning Representations (ICLR)*, May 2016.

[18] Y. Kim and A. M. Rush, "Sequence-level knowledge distillation," in *Proc. Empirical Methods in Natural Language Processing (EMNLP)*, Austin, TX, USA, Nov. 2016, pp. 1317–1327.

[19] A. Kuncoro, M. Ballesteros, L. Kong, C. Dyer, and N. A. Smith, "Distilling an ensemble of greedy dependency parsers into one MST parser," in *Proc. Empirical Methods in Natural Language Processing (EMNLP)*, Austin, TX, USA, Nov. 2016, pp. 1744–1753.

[20] A. Deoras, T. Mikolov, S. Kombrink, M. Karafiát, and S. Khudanpur, "Variational approximation of long-span language models for lvcsr," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Prague, Czech Republic, May 2011, pp. 5532–5535.

[21] A. Deoras, T. Mikolov, S. Kombrink, and K. Church, "Approximate inference: A sampling based modeling technique to capture complex dependencies in a language model," *Speech Communication*, vol. 55, no. 1, pp. 162–177, 2013.

[22] J. Goodman, "Classes for fast maximum entropy training," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 1, Salt Lake City, UT, USA, May 2001, pp. 561–564.

[23] R. Kneser and H. Ney, "Improved backing-off for m-gram language modeling," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Detroit, MI, USA, May 1995, pp. 181–184.

[24] Z. Tüske, W. Michel, R. Schlüter, and H. Ney, "Parallel neural network features for improved tandem acoustic modeling," in *Proc. Interspeech*, Stockholm, Sweden, Aug. 2017.

[25] R. Kneser and H. Ney, "Forming word classes by statistical clustering for statistical language modelling," in *Proc. First Int. Conf. on Quantitative Linguistics (QUALICO)*, Trier, Germany, 1991, pp. 221–226.

[26] M. Sundermeyer, R. Schlüter, and H. Ney, "rwthlm the RWTH Aachen University neural network language modeling toolkit," in *Proc. Interspeech*, Singapore, Sep. 2014, pp. 2093–2097.

[27] M. Sundermeyer, Z. Tüske, R. Schlüter, and H. Ney, "Lattice decoding and rescoring with long-span neural network language models," in *Proc. Interspeech*, Singapore, Sep. 2014, pp. 661–665.