INVESTIGATION ON LOG-LINEAR INTERPOLATION OF MULTI-DOMAIN NEURAL NETWORK LANGUAGE MODEL

Zoltán Tüske, Kazuki Irie, Ralf Schlüter, Hermann Ney

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

ABSTRACT

Inspired by the success of multi-task training in acoustic modeling, this paper investigates a new architecture for a multi-domain neural network based language model (NNLM). The proposed model has several shared hidden layers and domain-specific output layers. As will be shown, the log-linear interpolation of the multi-domain outputs and the optimization of interpolation weights fit naturally in the framework of NNLM. The resulting model can be expressed as a single NNLM. As an initial study of such an architecture, this paper focuses on deep feed-forward neural networks (DNNs). We also re-investigate the potential of long context up to 30-grams, and depth up to 5 hidden layers in DNN-LM. Our final feed-forward multi-domain NNLM is trained on 3.1B running words across 11 domains for English broadcast news and conversations large vocabulary continuous speech recognition task. After log-linear interpolation and fine-tuning, we measured improvements in terms of perplexity and word error rate over the models trained on 50M running words of in-domain news resources. The final multi-domain feed-forward LM outperformed our previous best LSTM-RNN LM trained on the 50M in-domain corpus, even after linear interpolation with large count models.

***Index Terms*—** multi-domain, language modeling, deep feed-forward network, LM adaptation, log-linear, interpolation

1. INTRODUCTION

The language model (LM) is a crucial component for various language and speech processing systems to achieve state-of-the-art results. Mainly, two approaches to train LMs are used and combined to build the best LM. First, the conventional count models, whose estimation is based on the relative frequencies of n-gram counts and a smoothing technique [1, 2]. Second, the neural network based language models (NNLMs), whose architecture can be a feed-forward [3, 4] or a recurrent network (RNN) [5]. Nowadays, NNLMs, especially recurrent ones, have become very popular and recent advances in LMs are mainly due to improvement in NN based language modeling, e.g. by introducing long short-term memory (LSTM) cells. As pointed out in [6], the potential of the deep feed-forward networks for LM are rarely investigated thoroughly. Due to the high computational cost, the NNLMs are often trained only on a relatively small selection of in-domain data set. Though some strategies have been studied to train NNLMs on large data [7, 8], only few works have been done using domain adaptation techniques for NNLMs. In fact, the domain adaptation for NNLMs does not have a straightforward solution as for the count model. In case of count models, the most common approach to train a domain adapted LM, when a large amount of data is available from different sources, is to train a LM separately on each sub-corpus. Then they are linearly interpolated with weights which are optimal to minimize the perplexity of the combined model on a given development data. By model architecture, the count models are suited to be linearly combined into one single model. Such an approach is possible for NNLMs, but turns out to be awkward because there is no straightforward manner to linearly combine individual NNLMs into one single model in the end.

Therefore, we study a new architecture for multi-domain NNLM which is inspired by multi-task training [9]. We show that if a common lexicon, shared hidden layers, and domain-specific final linear layers are applied, the combination of the multiple domain outputs naturally fits to the log-linear interpolation of [10]. Besides, the performance of DNN LMs being provided with long contexts (up to 30-gram) and depth (up to 5 non-linear layers) is also re-investigated.

The paper is organized as follows. After the overview of the related work in Section 2, Section 3 presents the log-linear interpolation of multi-domain NNLM. The details about our experimental setups are given in Section 4, and Section 5 presents the experimental results. The paper closes with conclusions in Section 6.

2. RELATED WORKS

Previous works on multi-domain adaptation of NNLMs for speech recognition include [11, 12] for feed-forward LMs and [13, 14] for RNN LMs. [11] investigated an unsupervised 2-pass approach: after a rescoring with an unadapted NNLM, an adaptation layer was inserted between projection and hidden layer. Whereas in [12], a domain dependent element-wise multiplication layer is set between projection and hidden layers of NNLM and learned during training. In [13], a domain dependent additive term is added to the RNN LM after the recurrent layer. On the other hand, in [14], the domain information is fed as an additional input feature to the RNN LM. The technique presented in this paper differs from these previous works as follows. In terms of model architecture, the investigated LM has domain dependent output layers inspired by the work of [9]. Besides linear interpolation of count model and NNLM, this study proposes the application of log-linear interpolation method of [10] on multi-domain NNLMs which is shown to result in a single model. As an initial step towards NNLM with multi-domain output, we focus only on deep feed-forward neural networks – like [15]. We also compare our model with the previous best LSTM-RNN LM of [16].

3. LOG-LINEAR INTERPOLATION OF NNLMS

In this work we limited our investigation on language models with conditional dependence on the previous $n-1$ words. However, the log-linear interpolation of multi-domain NN language model can be

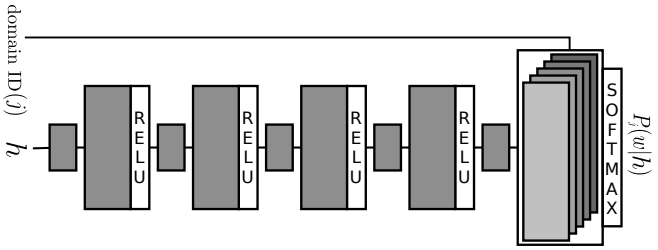

Fig. 1. *Multi-domain training of neural network LM. Except the last linear layer, the hidden layers are shared between the different domains.*

formulated in a general way using arbitrary history h. The log-linear interpolation of language models was introduced in [10]:

$$p(w|h) = \frac{1}{Z_\lambda} \prod_j p_j(w|h)^{\lambda_j}$$
$$Z_\lambda = \sum_w \prod_j p_j(w|h)^{\lambda_j} \quad (1)$$

Where $p(w|h)$ is the word posterior probability estimate after the interpolation, the scalar λ_j corresponds to the interpolation weight, $p_j(w|h)$ is the word posterior estimate of the jth model given the history. The Z_λ ensures that the output of the interpolated models sums up to one.

The last layer of a neural network corresponds to a log-linear model with linear feature functions. Assuming multiple log-linear models are trained on the same feature vector $y(h)$, e.g. extracted through a set of common nonlinear transformations from h (see Fig. 1), their log-linear combination results in the followings:

$$\prod_j p_j(w|h)^{\lambda_j} = \frac{\prod_j exp(\lambda_j(a_{wj}^T \cdot y + b_{wj}))}{\prod_j \sum_{w'} exp(\lambda_j(a_{w'j}^T \cdot y + b_{w'j}))} \quad (2)$$

Where vector a_{wj} and scalar b_{wj} generate the posterior estimate of word w of the jth model given y. In this paper different j corresponds to different domain. The common nonlinear transformations – e.g. shared deep feed-forward neural network layers – and the domain-dependent layer are trained jointly (Fig. 1). Substituting Eq. 2 into Eq. 1 results in

$$p(w|h) = \frac{exp(\tilde{a}_w^T \cdot y + \tilde{b}_w)}{\sum_{w'} exp(\tilde{a}_{w'}^T \cdot y + \tilde{b}_{w'})}$$
$$\text{where} \quad \tilde{a}_w = \sum_j \lambda_j \cdot a_{wj} \quad \text{and} \quad \tilde{b}_w = \sum_j \lambda_j \cdot b_{wj} \quad (3)$$

Eq. 3 indicates an advantageous property of a neural network with domain dependent last layer and shared hidden layers. The log-linear interpolation results in a single neural network comprising a weighted sum of the domain dependent linear layers. Furthermore, the implementation of log-linear interpolation is straightforward with existing neural network toolkits. First, the domain dependent weight matrices and biases should be interleaved row-wise. Denoting the lexicon size by W, the number of models by J, this step results in a huge weight matrix and bias vector which have $W \cdot J$ rows, e.g. 128k·11 in this study. Second, an additional linear layer – the interpolation layer – should be inserted before the softmax function. During the forward, the output vector of the merged linear layer should be re-interpreted as a matrix in column-major format with J rows and W columns. Thus, the interpolation layer should perform W times non-overlapping convolution. Updating only the interpolation weights, the optimization is a convex problem, and the resulting model cannot perform worse than the best fitting domain output. Due to the limited number of interpolation parameters the optimization can be carried out on a small e.g. development set similar to linear interpolation.

Table 1. *English text resources collected within the Quaero project, and count-based (KN4) LM perplexities (PPL) measured on the development set. cna:Central News Agency of Taiwan, English Service. ltw:Los Angeles Times/Washington Post Newswire Service, nyt:New York Times Newswire Service*

corpus		#words	interp. weights	KN4 PPL
Giga-word	cna	25M	0.0003	858.0
	ltw	221M	0.03	278.2
	nyt	1.1B	0.08	267.3
IWSLT 2013	lmtrain	2M	0.03	299.5
	train	2M	0.06	312.9
WMT 2012	news-crawl	914M	0.02	4969
Quaero	train10 - blog	149M	0.11	218.8
	train10 - news	153M	0.10	217.3
	train11	2M	0.37	215.8
	The Independent	508M	0.17	211.6
TED		2M	0.03	302.3
Interpolation		3.1B		132.7

4. EXPERIMENTAL SETUPS

Our experimental investigation on multi-domain feed-forward NNLM was performed on an English broadcast news and conversation speech recognition task from the Quaero project [17].

4.1. Acoustic models (AM)

For acoustic modeling, a hybrid 12-layer rectified linear unit (ReLU) activation based feed-forward multi layer perceptron (MLP) NN was trained [18, 19]. The network was built on vocal tract length normalized 50-dimensional Gammatone features [20]. In the first step, the AM was multilingually trained on 4 languages (French, English, German, Polish), and the amount of data totals up to 800 hours of speech. This initial training step, except for the model size, is analogous to [21], see there for more details. The multilingually boosted acoustic model was then fine-tuned with the 250 hours of available English target data. In the final step, the model was sharpened by the application of the minimum phone error sequence level discriminative training criterion [22]. The obtained model corresponds to our current best speaker independent AM trained for this task.

4.2. Baseline language models (LM)

Table 1 summarizes the different resources collected within the Quaero project to train the LM for English ASR. The corpora used in this study are the same as in [16]. Besides the transcription

Table 2. *Optimizing the length of history for feed-forward MLP LM on the 50M corpus. Perplexity measured on development set.*

N-gram	5	10	20	30
PPL	142.9	126.0	**117.4**	118.3

of acoustic data (*train11*), a substantial amount of text data was downloaded from web blogs and news [23]. After the clean-up and normalization of the text data, the lexicon was limited to the most frequent 150k words. On each text source, a 4-gram Kneser-Ney-smoothed LM was estimated (KN4). The LMs were then linearly interpolated minimizing the perplexity (PPL) on the development set. The development and the two test sets corresponds to the 2012, 2013, and 2011 ASR evaluation sets, respectively. Table 1 shows the individual and the interpolated model perplexities as well as the linear interpolation weight for each domain-specific model. The very high perplexity on WMT2012 is due to the mismatch in vocabulary coverage between the development data and this sub-corpus. Word-level perplexities are very heterogeneous for this sub-LM: better than the Quaero-train11 for some words but very bad for others.

4.3. Neural network LMs

Initial experiments were carried out on a smaller corpus which contained 50M running words. It was generated by including the in-domain data first (*train11*, 2M running words), and then adding more data from the second most relevant text source, as explained in [16].

During the NNLM training with the 50M corpus, the best matching 2M subset was placed at the end of the epoch. Instead of the full lexicon, only words occurring in the 50M corpus were considered for NNLM, in accordance with [16]. This resulted in a lexicon size of 128k. The 2M and 50M subsets were also used to fine-tune our NNLMs with in-domain data when the models were trained on the 3B corpus.

In this paper only ReLU activation units are used in the MLPs. Preliminary experiments showed that ReLU allowed larger mini-batch size, thus faster training, without any degradation in perplexity. In our network, we differentiate between three types of bottleneck (BN) layer. The *input BN* layer corresponds to the projection layer shared between the spliced sparse one-of-c vectors (LM history). In acoustic modeling this corresponds to a time-delay neural network layer [24]. *Between-hidden-layer BN* is applied to low-rank factorize the output of the hidden layers. In this study, the feed-forward NNLMs are trained *without* word classes, the posteriors of 128k words were estimated directly: an *output BN* was inserted before the last weight matrix to reduce the computation time. Although in acoustic modeling momentum and l_2 regularization terms are crucial for better generalization with ReLUs, for MLP based language modeling we did not observe severe overfitting by discarding them.

Estimating word posterior probability given the fixed length of history, the feed-forward NNLMs were optimized w.r.t. the cross-entropy criterion using stochastic gradient descent. In order to adjust the learning rate, the development set was used for cross validation (CV). Our conservative *newbob* strategy did not start to halve the learning rate until the CE objective function (log. perplexity) improved at least by 0.001. Furthermore, the ramping state was reset if the improvement was larger than 0.001. The training was stopped when the CE improvement was less than 0.0001 three times after each other in ramping state.

During model training or fine-tuning on the 2M or 50M subset, the learning rate scheduling was called epoch-wise. In order to train models on 3B running words, cross-validation was carried out after training on a shifting subset of 100M samples in the initial steps. Whenever the learning rate was reduced, the size of this subset was increased by a factor of 1.17. Performing 30 iterations within 3 epochs, the subset size thus was increased exponentially up to 300M training samples.

The training of our randomly initialized larger model (see Section 5.1) on 50M running words took about 3.5 days on a GTX980 GPU. The model training on 3B samples converged in 20 days using a single GPU. The training time between the multi-domain and classic MLP training does not differ substantially. The log-linear interpolation of the multi-domain models finished within 5 hours. The fine-tuning with the 50M corpus was always initiated with reduced learning rate and needed 1.5 day until convergence.

5. EXPERIMENTAL RESULTS

5.1. Optimizing the feed-forward MLP

A set of experiments was carried out to optimize the structure of our feed-forward NNLM, and to re-investigate also the potential of long context and depth with MLPs. To obtain the optimal context length, the following MLP structure was trained: the projection, between-hidden, and before-output BN layers contained 64, 256, and 128 nodes respectively. The MLP had 3 non-BN hidden layers with 1024 units each. The neural network was trained on the smaller, 50M matched training set. As can be seen in Table 2, training a 20-gram MLP LM resulted in the lowest perplexity on the development set.

In the second set of experiments, the structure of the MLP was optimized. As Rows 1, 2 and 6, 8 of Table 3 show, increasing the depth improved the perplexity about 1% relative. Increasing the depth up to 5 layers did not result in further gain (Rows 8, 9). Doubling the projection and output BN layer size resulted in over 2 point absolute PPL improvement (Rows 1, 3). Increasing the non-BN size showed another 1 point PPL gain (Rows 5, 6 and 7, 8). Discriminative pre-training (DPT) [25] was also found useful (Row 3, 5). An increased mini-batch size of 128 frames degraded the LM performance measurably (Row 3, 4). In summary, optimization of the MLP structure and context resulted in improved feed-forward LM performance, compared to previously reported results on this task. In [16], PPLs of 130.9 and 100.5 were reported for feed-forward NN and LSTM-RNN, respectively. It should also be noted that our model had only 58M parameters in contrast to the 160M of the LSTM-RNN, and the 63M n-grams of the 4-gram count model.

Table 3. *Parameter optimization of 20-gram feed-forward MLP LM on the 50M corpus. Perplexity measured on development set without interpolation with count model.*

non-BN		BN size			DPT	batch size	PPL	row
#	size	proj.	btw.hidden	output				
3	1024	64	256	128	-	64	117.4	1
5							116.2	2
3		128		256			114.7	3
						128	117.0	4
	2048				+	64	113.7	5
4	1024						112.1	6
	2048						111.5	7
5							**110.5**	8
							110.7	9

Table 5. *Quaero English speech recognition results. WER(%) are reported for Viterbi (Vi.) and confusion network decoding (CN).*

Language Model	Eval12 (dev)			Eval13			Eval11		
	PPL	Vi.	CN	PPL	Vi.	CN	PPL	Vi.	CN
KN4	132.7	12.6	12.3	131.1	10.7	10.5	133.4	15.4	15.0
+ 50M FFNN	96.5	11.4	11.1	97.2	9.6	9.5	95.0	14.2	13.8
+ 3B, fine-tune	89.6	10.9	10.7	90.6	9.4	9.1	88.0	**13.7**	**13.4**
+ Multi-domain,log-lin,fine-tune	**88.5**	**10.8**	**10.6**	**89.9**	**9.3**	9.1	**87.0**	**13.7**	13.5
+ LSTM	91.6	10.9	10.8	92.0	**9.3**	**9.0**	91.0	**13.7**	13.5

Table 4. *Effect of training the models on more data, fine-tuning with matched data, and initialization with log-linear interpolated and multi-domain style trained model. Perplexities reported after multi-domain training without interpolation corresponds to the use of the best output.*

LM	multi domain	log.lin interp.	fine tuning 50M	fine tuning 2M	PPL	row
50M					110.5	1
				×	109.0	2
3B					129.0	3
				×	96.6	4
			×		101.4	5
			×	×	96.2	6
	×				133.1	7
	×		×	×	95.7	8
	×	×			117.6	9
	×	×	×	×	**94.3**	10

5.2. Training MLP-LMs on multi-domain data

The optimized MLP was then used to investigate the multi-domain training of NNLMs. First, a LM without domain dependent output was built. As can be seen in Table 4, training the model on more but mismatched data did not help alone (Rows 1, 3). However, fine-tuning on a small, 2M running words, set of domain-specific data led to over 30 point absolute PPL improvement (Rows 1, 4). The results indicate that the MLP LM training is a difficult optimization problem and mismatched data can help to avoid local optima, see e.g. results in Section 5.1. Although the 50M set contained the 2M (see Section 4.3) subset, for a fair comparison the 50M model was also tuned with multiple epochs on the 2M set. This step improved the 50M model by 1% relative (Rows 1, 2). Fine-tuning the 3B model on 50M and 2M sets in a fixed sequential order led only to slightly better perplexities (Rows 4, 6).

Switching on the domain dependent layer, we obtained 11 different perplexities. Perplexity measured on the best fitting output (corresponding to *Quaero - The Independent*) is shown in Row 7, and after fine-tuning in Row 8. Performing the log-linear interpolation of the outputs improved the result over the 3B baseline significantly (Rows 3, 9), but it still did not reach the performance of the in-domain 50M model. However, we obtained the best model if the log-linearly interpolated model was used as initialization in the fine-tuning step (Row 10). The log-linear interpolation was also compared with linear interpolation. The later one performed slightly better achieving a PPL of 114. Nevertheless, fine-tuning of such a model is more sophisticated, and the multi-domain weights cannot be merged into a simple NNLM, the interpolated model ends up in an increased number of parameters compared to the other 3B-word models in Rows 3 or 10. In summary, with the help of multi-domain data our MLP-LM achieved 94.3 PPL, 15% rel. improvement over the 50M in-domain model. The best result in Tabel 4 shows 6% relative improvement – with much less parameters – over the previous best PPL result of 100.5 achieved by a LSTM-LM in [16].

In the final experiment our feed-forward LMs were tested in broadcast news and conversations speech recognition task. We used the RASR [26] software for lattice extraction, Viterbi (Vi) and confusion network (CN) decoding. Applying the traceback approximation of [27], lattices were rescored by the `rwthlm` [28] toolkit. Analogously to [16] the search space is initially generated with the count LM trained on 3B running words. Due to the improved acoustic model the count model baseline result is already as good as the best results reported in [16]. The lattice rescoring was carried out with neural network LM linearly interpolated with the large count LM in all cases. Table 5 shows perplexity (PPL) and word error rate (WER) results on three different evaluation corpora of the Quaero project (Eval11, Eval12, Eval13). As in [16], the Eval12 set was used as development set to tune all the interpolation weights, LM scale, and also as CV set during NNLM training. Results are also reported on the Eval11 set in order to allow WER comparison with our previous works made on acoustic modeling, e.g. [29, 21]. However, in this work, an unpruned count LM was directly used for decoding accounting for $\sim$0.3% absolute WER improvement over the pruned model.

As can be seen, our best 50M feed-forward neural network (FFNN) LM is only 5 points in PPL (rel. 5%) behind the LSTM model. Training an FFNN model on more multi-domain data and then fine-tuning with in-domain data resulted in significantly lower WER. Our best FFNN is 2-4 points in PPL better than our previous best LSTM. However, this improvement did not always carry over into significantly better recognition results after CN decoding.

6. CONCLUSIONS

In this work, feed-forward deep neural network LMs are shown to obtain significantly better performance than previously reported. Further gain was achieved by avoiding local optima using a large amount of mismatched out-of-domain text resources to initialize the model before fine-tuning on matched data. We also demonstrated that multi-domain training and log-linear interpolation of domain-specific models could result in a single neural network model allowing even better initialization before adaptation. Systematic application of these techniques led to high performing feed-forward NNLM showing similar or better results than our current best LSTM models.

In future work, multi-domain training will also be repeated with LSTMs. For better comparison with our proposed FFNN models, further investigation on LSTM with full-output and experiments with larger data sets are also necessary. Aiming at better log-linear interpolation of NNLMs, the training of multi-domain layers on the output of hidden activations of in-domain sub-networks might also be considered.

7. REFERENCES

[1] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," *Computer Speech & Language*, vol. 13, no. 4, pp. 359–393, 1999.

[2] J. T. Goodman, "A bit of progress in language modeling," *Computer Speech & Language*, vol. 15, no. 4, pp. 403–434, 2001.

[3] Y. Bengio, R. Ducharme, and P. Vincent, "A neural probabilistic language model," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 13, Denver, CO, USA, 2000, pp. 932–938.

[4] H. Schwenk, "Continuous space language models," *Computer Speech & Language*, vol. 21, no. 3, pp. 492–518, 2007.

[5] T. Mikolov, S. Kombrink, L. Burget, J. H. Cernocky, and S. Khudanpur, "Extensions of recurrent neural network language model," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Prague, Czech Republic, May 2011, pp. 5528–5531.

[6] H. Schwenk, F. Bougares, and L. Barrault, "Efficient training strategies for deep neural network language models," in *NIPS Workshop on Deep Learning and Representation Learning*, Montreal, Canada, Dec. 2014.

[7] H. Schwenk and J.-L. Gauvain, "Training neural network language models on very large corpora," in *Proc. of the conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT/EMNLP)*, Vancouver, Canada, Oct. 2005, pp. 201–208.

[8] T. Mikolov, A. Deoras, D. Povey, L. Burget, and J. Černockỳ, "Strategies for training large scale neural network language models," in *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, Waikoloa, HI, USA, Dec. 2011, pp. 196–201.

[9] R. Caruana, "Multitask learning: A knowledge-based source of inductive bias," in *Proc. of International Conference on Machine Learning (ICML)*, Amherst, MA, USA, Jun. 1993, pp. 41–48.

[10] D. Klakow, "Log-linear interpolation of language models." in *Proc. of International Conference on Spoken Language Processing (ICSLP)*, Sydney, Australia, 1998.

[11] J. Park, X. Liu, M. J. Gales, and P. C. Woodland, "Improved neural network based language modelling and adaptation," in *Proc. Interspeech*, Makuhari, Japan, Sep. 2010, pp. 1041–1044.

[12] T. Alumäe, "Multi-domain neural network language model." in *Proc. Interspeech*, Lyon, France, Aug. 2013, pp. 2182–2186.

[13] O. Tilk and T. Alumäe, "Multi-domain recurrent neural network language model for medical speech recognition," in *Proc. of the conference on Human Language Technologies (HLT)*, vol. 268, 2014, pp. 149–152.

[14] X. Chen, T. Tan, X. Liu, P. Lanchantin, M. Wan, M. J. Gales, and P. C. Woodland, "Recurrent neural network language model adaptation for multi-genre broadcast speech recognition," in *Proc. Interspeech*, Dresden, Germany, 2015, pp. 3511–3515.

[15] E. Arisoy, T. N. Sainath, B. Kingsbury, and B. Ramabhadran, "Deep neural network language models," in *Proc. of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*, Montréal, Canada, Jun. 2012, pp. 20–28.

[16] M. Sundermeyer, H. Ney, and R. Schlüter, "From feedforward to recurrent lstm neural networks for language modeling," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 3, pp. 517–529, Mar. 2015.

[17] M. Nußbaum-Thom, S. Wiesler, M. Sundermeyer, C. Plahl, S. Hahn, R. Schlüter, and H. Ney, "The RWTH 2009 QUAERO ASR evaluation system for English and German," in *Proc. Interspeech*, Makuhari, Japan, Sep. 2010, pp. 1517–1520.

[18] H. A. Bourlard and N. Morgan, *Connectionist Speech Recognition: A Hybrid Approach*. Norwell, MA, USA: Kluwer Academic Publishers, 1993.

[19] S. Scanzio, P. Laface, L. Fissore, R. Gemello, and F. Mana, "On the Use of a Multilingual Neural Network Front-End," in *Proc. Interspeech*, Brisbane, Australia, Sep. 2008, pp. 2711–2714.

[20] R. Schlüter, I. Bezrukov, H. Wagner, and H. Ney, "Gammatone features and feature combination for large vocabulary speech recognition," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Honolulu, HI, USA, Apr. 2007, pp. 649–652.

[21] Z. Tüske, R. Schlüter, and H. Ney, "Multilingual Hierarchical MRASTA Features for ASR," in *Proc. Interspeech*, Lyon, France, Aug. 2013, pp. 2222–2226.

[22] D. Povey and P. Woodland, "Minimum phone error and i-smoothing for improved discriminative training," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Orlando, FL, USA, May 2002, pp. I–105–I–108.

[23] M. Sundermeyer, M. Nußbaum-Thom, S. Wiesler, C. Plahl, A.-D. Mousa, S. Hahn, D. Nolden, R. Schlüter, and H. Ney, "The RWTH 2010 QUAERO ASR Evaluation System for English, French, and German," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Prague, Czech Republic, May 2011, pp. 2212–2215.

[24] A. Waibel, T. Hanazawa, G. Hinton, S. Kiyohiro, and K. J. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 37, no. 3, pp. 328–339, 1989.

[25] F. Seide, G. Li, X. Chen, and D. Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transcription," in *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, Waikoloa, HI, USA, Dec. 2011, pp. 24–29.

[26] D. Rybach, S. Hahn, P. Lehnen, D. Nolden, M. Sundermeyer, Z. Tüske, S. Wiesler, R. Schlüter, and H. Ney, "RASR - The RWTH Aachen University Open Source Speech Recognition Toolkit," in *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, Waikoloa, HI, USA, Dec. 2011.

[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.

[28] 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.

[29] S. Wiesler, A. Richard, R. Schlüter, and H. Ney, "Mean-normalized stochastic gradient for large-scale deep learning," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Florence, Italy, May 2014, pp. 180–184.