Optimizing Deep Bottleneck Feature Extraction

Quoc Bao Nguyen, Jonas Gehring, Kevin Kilgour and Alex Waibel
International Center for Advanced Communication Technologies - InterACT,
Institute for Anthropomatics, Karlsruhe Institute of Technology, Germany
e-mail: quoc.nguyen@kit.edu, jonas.gehring@kit.edu

***Abstract*—We investigate several optimizations to a recently published architecture for extracting bottleneck features for large-vocabulary speech recognition with deep neural networks. We are able to improve recognition performance of first-pass systems from a 12% relative word error rate reduction reported previously to 21%, compared to MFCC baselines on a Tagalog conversational telephone speech corpus. This is achieved by using different input features, training the network to predict context-dependent targets, employing an efficient learning rate schedule and varying several architectural details. Evaluations on two larger German and French speech transcription tasks show that the optimizations proposed are universally applicable and yield comparable gains on other corpora (19.9% and 22.8%, respectively).**

I. Introduction

Bottleneck features (BNFs) obtained from multi-layer perceptrons (MLPs) have become an inherent part in many automatic speech recognition systems. This success is due to their discriminative power and robustness regarding speaker and environment variations. In the standard setup as proposed by Grézl [1], the MLP has three hidden layers. One of those layers is typically very small (the bottleneck) and provides the final features that can be used by Gaussian Mixture models for phoneme state estimation.

Recently, deep learning algorithms that deal with training deep neural networks (DNNs) consisting of many hidden layers have been successfully applied to many signal processing tasks, including computer vision [2] and acoustic modeling [3]. A popular approach is to pre-train individual layers as restricted Boltzmann machines (RBMs), which are unsupervised generative models [4]. Ideally, the pre-training procedure initializes the network parameters in a space that is beneficial for subsequent supervised training towards the actual classification task [5]. It has also been shown that other network models such as auto-encoders can be used for pre-training as well [6].

A natural extension to extract better bottleneck features is thus the replacement of the standard MLP consisting of three hidden layers with a deep neural network. This was first attempted in 2011 by Yu and Seltzer, which embedded the bottleneck as a small RBM in the middle of a deep network [7]. After pre-training the network layer by layer on windows of MFCC features, the whole network was trained to estimate either context-independent or context-dependent HMM target states. They found that pre-training produced better features, and that the use of context-dependent targets was helpful as well. Their architecture could not make use of more than 5 hidden layers, which was presumably caused by placing the bottleneck layer in the middle of the network.

A different approach has been proposed by Sainath et al. [8], in which the bottleneck is placed in an auto-encoder network trained on HMM state posterior probabilities estimated by a separate deep neural network. As they added more hidden layers to the DNN, they obtained a better prediction of phonetic states and consequently better bottleneck features extracted by the auto-encoder network.

Gehring et al. recently introduced an architecture that consists of a single network and that is able to exploit the increased modeling power of deep networks [9]. When extracting bottleneck features from raw log mel scale filterbank coefficients with sufficiently large and pre-trained networks, they could significantly out-perform several MFCC baseline systems. They stated that in their experiments, mel scale features generally worked better than MFCCs but did not perform further feature engineering or task-specific architectural optimizations. Furthermore and in contrast to [7], they did not use context-dependent targets for supervised network training.

Regarding input feature optimization for BNF networks, Kilgour et al. recently presented a study regarding warped Minimum Variance Distortionless Response (wMVDR) features [10]. They investigated the effect of using different features for the neural network input, and found that combining MFCC and wMVDR features resulted in improved bottleneck features. Plahl et al. also investigated several input feature combinations [11], and reported gains by merging MFCC, PLP and Gammatone filter outputs at the network input compared to performing a system combination from lattices generated by BNF systems trained on the individual features. None of these works incorporated deep learning techniques, though.

In this work, we try to apply several recently proposed optimizations for bottleneck feature extraction to the architecture proposed by Gehring et al [9]. We show that as in [10], improvements can be obtained by combining MFCC and wMVDR input features and that using context-dependent targets generally produces better features. By scheduling the learning rate for supervised fine-tuning, we are able to lower final word error rates as well as the total required training time. We also show that adjusting to the size of the hidden layers (excluding the bottleneck layer) can yield additional gains.

II. Deep Bottleneck Features

In this section, we briefly describe the deep neural network architecture for bottleneck feature extraction proposed in [9] and depicted in Figure 1. The network consists of a variable number of moderately large, fully connected hidden layers and a small bottleneck layer which is followed by an additional hidden layer and the final classification layer. The architecture differs from setups previously described, where the bottleneck

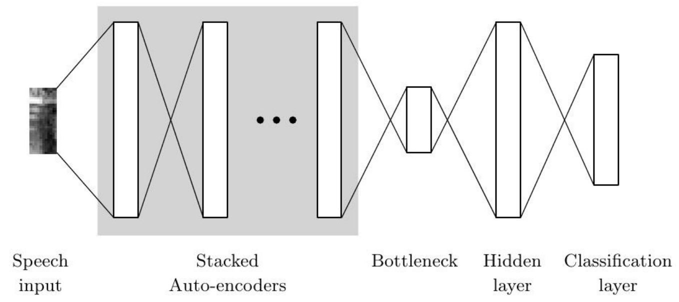

Fig. 1: Deep network architecture propsed by Gehring et al. [9]

layer has been placed in the middle of a deep network [7], [12] or added as a second model trained on the output values of the original network [8].

The hidden layers in front of the bottleneck are initialized using unsupervised, layer-wise pre-training. Thanks to their success in the deep learning community, restricted Boltzmann machines have become the default choice for pre-training the individual layers of deep neural networks used in speech recognition. Gehring et al. demonstrated that denoising auto-encoders [13], which are straight-forward models that have been successfully used for pre-training neural architectures for computer vision and sentiment classification [14], are applicable to speech data as well.

We follow their training scheme and initialize the hidden layers as denoising auto-encoders, too. Like regular auto-encoders, these models consist of one hidden layer and two identically-sized layers reprsenting the input and output values. The network is usually trained to reconstruct its input at the output layer with the goal to generate a useful intermediate representation in the hidden layer. In denoising auto-encoders, the network is trained to reconstruct a randomly corrupted version of its input, which can be interpreted as a regularizing mechanism that facilitates the learning of large and over-complete hidden representations [13].

For denoising auto-encoders working on binary data (i.e. grayscale images or sigmoid activations of a previous hidden layer), Vincent et al. proposed the use of masking noise for corrupting each input vector [13]. Every element of the input vector is randomly set to zero with a fixed probability (we settled with 20% here). The cross-entropy error objective $L_H(x,z) = \sum_i x_i \log z_i + (1 - x_i) \log z_i$ is then used to compare the reconstructed output with the original, uncorrupted input in order to obtain the gradients necessary for adjusting the network weights. When training a network on speech features like MFCCs, the first layer models real-valued rather than binary data, so the mean squared error $L_2(x,z) = \sum_i (x_i - z_i)^2$ is selected as the training criterion. In this work, we also apply masking noise to the first layer, although other types of noise could be used as well [13].

After a stack of auto-encoders has been pre-trained in this fashion, a deep neural network can be constructed. The bottleneck layer, an additional hidden layer and the classification layer are initialized with random weights and connected to the hidden representation of the top-most auto-encoder. While all out hidden units use sigmoid non-linearities, the classification layer output is obtained with the softmax activation function. The resulting network is then trained with supervision to estimate either context-independent or context-dependent HMM tri-phone states. For this last training step, errors are obtained with the cross-entropy function. Finally, the last two layers of the network can be discarded as the units in the bottleneck layer provide the final features used for training standard Gaussian mixture (GMM) acoustic models.

III. Baseline Systems

A. Corpora Description

We performed experiments on several datasets that differ in language as well as speaking style and recording condition. For tuning the bottleneck feature extraction, we worked with a challenging Tagalog conversational speech corpus. This dataset was released in 2012 under the identifier “babel106-v0.2f” in the IARPA BABEL program [15]. It contains 79 hours of narrow-band speech, of which 69 were used to train the feature extraction networks and acoustic models. The numbers reported were obtained by decoding the remaining 10 hours.

The best architectures were also evaluated on two larger Quaero datasets containing broadcast news speech in German and French. These corpora were released between 2010 and 2012 and contain 188 and 267 hours of wide-band speech, respectively.

B. Baseline Systems

Baseline system training and decoding was performed with the Janus Recognition Toolkit (JRTk) developed at Karlsruhe Institute of Technology and Carnegie Mellon University [16]. For the baseline, samples consisting of 13 MFCCs were extracted from the audio signal with a frame shift of 10 ms and stacked with 15 adjacent samples. This resulted in feature vectors consisting of 195 elements. LDA was applied to reduce those to 42 dimensions, which constituted the final input features for the recognition system. Acoustic model training was performed in a context-dependent setup with three states per phoneme, and a left-to-right topology without skip states.

For Tagalog, all models used 10000 clustered HMM states and were trained using incremental splitting of Gaussians (MAS) training, followed by optimal feature space and Viterbi training. The German and French setups are similar, but 6000 and 8000 HMM states were used here, respectively. Furthermore, the broadcast news systems made use of vocal tract length normalization (VTLN).

Bottleneck features were extracted from different features as described below. Unless otherwise noted, context windows were constructed by concatenating the feature vectors of 11 neighboring samples. Each of the hidden layers contained 1000 units, while 42 units were placed in the bottleneck layer and 149 context-independent target states were used for supervised training on the Tagalog dataset. Six pre-trained auto-encoder layers were placed in front of the bottleneck layer. The acoustic model training for BNF systems was identical to the baseline systems described above.

IV. Optimizations

In this section, we describe and evaluate the individual optimizations proposed in order to improve the performance of the deep bottleneck feature (DBNF) extraction setup described previously.

First, we experimented with MFCC and wMVDR neural network input features as they have been successfully applied for shallow bottleneck networks before [10]. We then investigated whether the neural network can benefit from larger hidden layers. Third, we applied learning rate scheduling using the "newbob" schedule in order to speed up the network training procedure. Finally, we replaced the monophone targets used in the supervised training stage with context-dependent HMM states and tuned the number of hidden layers as well as the input context window size.

A. Input features

In Table I, we compare the recognition performance in word error rate (WER) of GMMs trained on DBNFs extracted from different features. We trained DBNF networks for the Tagalog corpus on 20 mel-frequency cepstrum cofficients (MFCC20), 40 log mel scale coefficients (lMEL40) and cepstral wMVDR coefficients [17] (wMVDR20). With MFCC and log mel features, a relative reduction in word error rate of 11.7% is obtained. This result is similar to the original experiment reported in [9] on the same corpus, where a reduction of 12.0% was achieved. With wMVDR features, the error rate can be decremented further to 59.4%. The combination of MFCC and wMVDR features was found to be helpful in [10] but resulted in slightly worse recognition performance compared to wMVDRs only in this experiment.

TABLE I: *Recognition performance for the Tagalog system with various input features*

Features	**WER (%)**
MFCC baseline	68.0
DBNF-MFCC20	60.0
DBNF-lMEL40	60.0
DBNF-wMVDR20	**59.4**
DBNF-MFCC20+wMVDR20	59.6

B. Learning Rate Scheduling

In [9], a small learning rate of 0.05 was used to train the neural network for 50 epochs once the auto-encoder layers had been pre-trained. In order to speed up the training procedure we evaluated the performance of DBNF networks trained with the "newbob" schedule, which is a popular choice for setting learning rates in the speech recognition community. An initial high learning rate is kept fixed as long as the increase in frame-level accuracy on a held-out validation set between successive epochs is higher than 0.5%. The learning rate is then halved for epoch until the validation accuracy drops below a second threshold (we used 0.01% here). At this point, the network training stops. As in [9], DBNF networks were trained with stochastic gradient descent on mini-batches, using a batch size of 256.

As can be seen in Table II, small improvements over the results in Table I were obtained (59.6% to 59.2% WER for the combination of MFCCs and wMVDRs). More importantly, the training time for the networks dropped from 50 to 15 epochs on average, which is a significant speed-up of 70%. Here, the combination of MFCCs and wMVDRs performs slightly better.

TABLE II: *Word error rate on Tagalog with "newbob" learning rate scheduling*

Features	**WER (%)**
DBNF-wMVDR20	59.3
DBNF-MFCC20+wMVDR20	59.2

The remaining experiments in this paper were performed with the "newbob" learning rate schedule only.

C. Architecture

We further examined how the size of the hidden layers (exluding the fixed-size bottleneck layer and the classification layer) impacts the final recognition performance. As is shown in Table III, decreasing the number of units in the hidden layers from 1000 as proposed in [9] to 800 increased the word error rate by 0.4% absolute, while increasing the number of units to 1200 reduced the WER to 59.0%. Adding more units does not help for this setup with 69 hours of training data (almost 25 million frames) and context-dependent monophone targets.

TABLE III: *Effects of varying the number of units in the hidden layers of the DBNF network*

Features	**Layer Size**	**WER (%)**
DBNF-MFCC+wMVDR	800	59.6
	1000	59.2
	1200	**59.0**
	1400	**59.0**

D. Context-Dependent Targets

The usage of context-dependent HMM target states for network training has significantly contributed to the recent success of deep neural network acoustic models [18] and has been found to work well for bottleneck features, too [7]. According to the results listed in Table IV, the deep bottleneck feature extraction scheme benefits from using senone targets as well: for the MFCC+wMVDR system with 4000 target states and 1000 units in the hidden layers, recognition performance could be increased by 5.3% relative to 55.9% WER when compared to the results obtained with context-independent targets shown in Table III. By increasing the number of context-dependent states to 10000, the error rate could be lowered to 54.4%. Further improvements were obtained by training networks with larger hidden layers. In contrast to the previous experiment, the network was able to make use of larger hidden layers with up to 16000 units (53.5% WER). Features extracted from

TABLE IV: *Resulting error rates when using context-dependent targets for network training*

Features	Targets	Layer Size	WER (%)
DBNF-MFCC+wMVDR	4000	1000	55.9
	10000	1000	54.4
	10000	1200	54.1
	10000	1400	53.8
	10000	1600	**53.5**
	10000	1800	53.8
DBNF-wMVDR	10000	1000	56.0

both MFCCs and wMVDRs outperformed those obtained from wMVDRs by 1.6% absolute when using 10000 states.

With the setup containing 1400 units per hidden layer and 10000 target states, we performed further architectural optimizations and varied the number of auto-encoder layers placed in front of the bottleneck. As shown in Table V, the best result could be achieved by using either 5 or 6 layers (resulting in a DBNF network with 8 or 9 layers, respectively), with no additional gains obtained by adding further layers.

TABLE V: *Error rates for DBNFs trained with context-dependent targets and different numbers of layers*

Layers	2	3	4	5	6	7
WER (%)	55.3	54.2	54.2	**53.8**	**53.8**	54.3

E. Context Window

The size of the context window that the DBNF extraction network is able to observe directly influences the frame-level accuracy during training. A larger context window increases the accuracy at which the network is able to predict the HMM state at a given sample. We thus varied the window size in order to investigate whether the improved accuracy would result in more useful bottleneck features. The numbers in Table VI were obtained using DBNFs trained on MFCCs and wMVDRs with 1400 units per hidden layer. Increasing the context window to 13 frames (130 ms) reduced the recognition error by 0.6% to 53.2% WER. Further enlargements resulted in worse recognition performance.

TABLE VI: *Influence of varying the size of the input context window*

Frames	11	13	15	17	19	21
WER (%)	53.8	**53.2**	53.9	53.9	53.9	54.2

F. General Applicability

The optimizations described above were performed on a relatively small corpus with a baseline that was among our early system builds. We thus evaluated a well-performing configuration against stronger baseline systems on larger datasets in order to check its general applicability.

For the German and French Quaero corpora, we trained networks observing 15 neighboring frames and 5 auto-encoder layers containing 1600 hidden units each. The same number of states as for the baseline systems were used to obtain errors to adjust the network parameters during supervised training. As can be seen in Table VII, the improvements obtained by optimizing the networks on Tagalog could be carried over to these setups. For the MFCC+wMVDR feature combination, comparable gains of 19.9% relative in German and 22.8% in French could be achieved.

TABLE VII: *Performance with optimized DBNF networks for larger broadcast news corpora*

Features	Language	Baseline WER (%)	DBNF WER (%)
MFCC+wMVDR	German	20.7	16.6
MFCC+wMVDR	French	25.1	19.4

V. Conclusion

In this work, we have evaluated several enhancements to a previously published scheme for extracting bottleneck features with deep neural networks. The largest increase in performance was obtained by training the DBNF network on a large number of context-dependent targets, followed by combining MFCC and wMVDR input features. The time required to train the neural networks with supervision could be reduced significantly by scheduling the learning rate with the "newbob" algorithm. Further gains were achieved by enlarging the hidden layers of the network and the context window of accessible input features.

The DBNF extraction was tuned on a medium-sized and challenging Tagalog conversation speech corpus, which increased the relative improvement in word error rate over the MFCC baseline from 11.8% to 21%. Evaluations on two larger corpora containing broadcast news speech demonstrated that the optimizations performed can be successfully applied to other tasks as well.

Since architectural optimizations change the number of trainable parameters in the neural network, they partially depend on the amount of data available for training. It might therefore be worthwhile to re-run certain optimizations on the two larger datasets in order to obtain even better bottleneck features. In the future, we would like to directly compare the performance of our DBNF extraction scheme with other network architectures as well as hybrid DNN/HMM systems. Furthermore, we are interested in integrating recently proposed hierarchical and multi-lingual network training approaches into our architecture.

Acknowledgments

Supported in part by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense

U.S. Army Research Laboratory (DoD/ARL) contract number W911NF-12-C-0015. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoD/ARL, or the U.S. Government.

This work was partly realized within the Quaero Programme, funded by OSEO, the French State agency for innovation.

References

[1] F. Grézl, M. Karafiát, S. Kontáir, and J. Cernocky, "Probabilistic and bottle-neck features for lvcsr of meetings," in *Acoustics, Speech and Signal Processing (ICASSP), 2007 IEEE International Conference on. IEEE*, 2007, pp. V–757 – IV–760.

[2] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, vol. 25, 2012, pp. 1106–1114.

[3] F. Seide, G. Li, and D. Yu, "Conversational speech transcription using context-dependent deep neural networks," in *Proc. Interspeech*, 2011, pp. 437–440.

[4] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets,," in *Neural computation*, vol. 18, 2006, pp. 1527–1554.

[5] D. Erhan, A. Courville, Y. Bengio, and P. Vincent, "Why does unsupervised pre-training help deep learning?" in *Proceedings of AISTATS 2010*, vol. 9, May 2010, pp. 201–208.

[6] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, U. D. Montréal, and M. Québec, "Greedy layer-wise training of deep networks," in *In NIPS*. MIT Press, 2007.

[7] D. Yu and M. L. Seltzer, "Improved bottleneck features using pretrained deep neural networks," in *INTERSPEECH*, 2011, pp. 237–240.

[8] T. Sainath, B. Kingsbury, and B. Ramabhadran, "Auto-encoder bottleneck features using deep belief networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, 2012, pp. 4153–4156.

[9] J. Gehring, Y. Miao, F. Metze, and A. Waibel, "Extracting deep bottleneck features using stacked auto-encoders," in *ICASSP2013*, Vancouver, CA, 2013, pp. 3377–3381.

[10] K. Kilgour, T. Seytzer, Q. Nguyen, and A. Waibel, "Warped minimum variance distortionless response based bottle neck features for lvcsr," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, 2013.

[11] C. Plahl, R. Schlüter, and H. Ney, "Improved acoustic feature combination for lvcsr by neural networks," in *INTERSPEECH*, 2011, pp. 1237–1240.

[12] Z. Tüske, R. Schlüter, and H. Ney, "Deep hierarchical bottleneck mrasta features for lvcsr," 2013.

[13] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *ICML08*, 2008, pp. 1096–1103.

[14] X. Glorot, A. Bordes, and Y. Bengio, "Domain adaptation for large-scale sentiment classification: A deep learning approach," in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 2011, pp. 513–520.

[15] Intelligence Advanced Research Projects Activity, "IARPA-BAA-11-02," http://www.iarpa.gov/Programs/ia/Babel/babel.html, 2011, last accessed July 16, 2013.

[16] H. Soltau, F. Metze, C. Fugen, and A. Waibel, "A one-pass decoder based on polymorphic linguistic context assignment," in *Automatic Speech Recognition and Understanding, 2001. ASRU '01. IEEE Workshop on*, 2001, pp. 214–217.

[17] M. Wölfel and J. McDonough, "Minimum variance distortionless response spectral estimation," *Signal Processing Magazine, IEEE*, vol. 22, no. 5, pp. 117–126, 2005.

[18] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 20, no. 1, pp. 30–42, 2012.