INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

Exploring the Encoder Layers of Discriminative Autoencoders for LVCSR

Pin-Tuan Huang[1], *Hung-Shin Lee*[1], *Syu-Siang Wang*[2], *Kuan-Yu Chen*[3], *Yu Tsao*[2], *Hsin-Min Wang*[1]

[1]Institute of Information Science, Academia Sinica, Taiwan
[2]Research Center for Information Technology Innovation, Academia Sinica, Taiwan
[3]National Taiwan University of Science and Technology, Taiwan

melody.hpt508@gmail.com, hungshinlee@gmail.com

Abstract

Discriminative autoencoders (DcAEs) have been proven to improve generalization of the learned acoustic models by increasing their reconstruction capacity of input features from the frame embeddings. In this paper, we integrate DcAEs into two models, namely TDNNs and LSTMs, which have been commonly adopted in the Kaldi recipes for LVCSR in recent years, using the modified nnet3 neural network library. We also explore two kinds of skip-connection mechanisms for DcAEs, namely concatenation and addition. The results of LVCSR experiments on the MATBN Mandarin Chinese corpus and the WSJ English corpus show that the proposed DcAE-TDNN-based system achieves relative word error rate reductions of 3% and 10% over the TDNN-based baseline system, respectively. The DcAE-TDNN-LSTM-based system also outperforms the TDNN-LSTM-based baseline system. The results imply the flexibility of DcAEs to be integrated with other existing or prospective neural network-based acoustic models.

Index Terms: discriminative autoencoders, time delay neural networks, recurrent neural networks, LVCSR

1. Introduction

Due to the introduction of deep neural networks (DNNs) and the fact that computing power has advanced by leaps and bounds over the past decade, automatic speech recognition (ASR) has gained more attention from various artificial intelligence (AI)-related fields and communities. DNNs have also become mainstream architectures for acoustic modeling in large vocabulary continuous speech recognition (LVCSR) instead of traditional GMM-HMM (hidden Markov model with Gaussian mixture model emissions) based models [1, 2]. What have followed naturally are many free-to-use toolkits supporting the usage of GPUs for building ASR systems, such as Microsoft's CNTK[1], RWTH Aachen University's RASR[2], and Kaldi developed primarily by Daniel Povey and his research team [3].

There is no doubt that Kaldi is one of the most popular open-source toolkits, providing industrial engineers and academic researchers working in the speech processing area with the most advanced and regularly updated training recipes and a fair ground for comparison. Based on the topology of HMMs and weighted finite-state transducers (WFSTs), Kaldi also generates high-quality word/phone lattices that are not only sufficiently efficient for real-time decoding but also more effective than other potential and promising end-to-end approaches, such as connectionist temporal classification (CTC) [4, 5, 6, 7, 8, 9, 10], sequence-to-sequence attention-based models [11, 12, 13], and their hybrid version [14].

[1]https://www.microsoft.com/en-us/cognitive-toolkit
[2]https://www-i6.informatik.rwth-aachen.de/rwth-asr

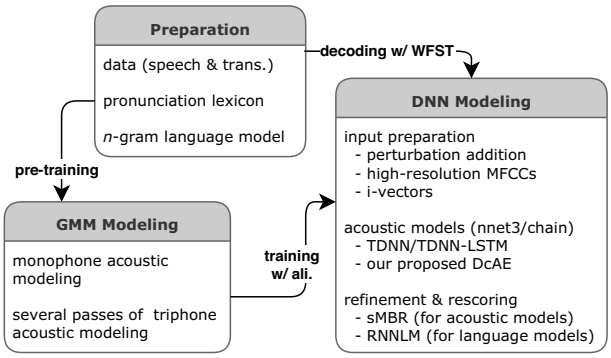

Figure 1: *Diagram of the standard GMM-DNN-HMM-based recipe of Kaldi adopted in this paper.*

In Kaldi, a standard recipe for GMM-DNN-HMM-based ASR generally consists of three steps, as shown in Figure 1. In the first step of preparation, users must provide a pronunciation lexicon and language models corresponding to the target language or output texts, and specify the speech utterances and their corresponding transcripts for training and evaluation. In this step, training data augmentation techniques, such as speed/volume perturbations [15] and multi-condition training with noise and reverberation additions [16, 17], can be considered. In the second step, based on the defined speech units (e.g., phonemes or sub-syllables), the GMM-HMM system is initialized by monophone model training and extended to a context-dependent triphone system. In the last step, a more accurate acoustic model is developed by DNNs based on the frame-level triphone alignment generated by the pre-trained GMM-HMM system. The acoustic model built up by DNNs is suitable for integrating higher-resolution acoustic features and i-vectors, which contain abundant speaker-related information [18].

As for DNN modeling, there are currently four setups in Kaldi, namely `nnet1`, `nnet2`, `nnet3`, and `chain`. The main difference between `nnet1` and `nnet2` is that the former uses DNN-generated alignment and mini-batch stochastic gradient descent (SGD), while the latter uses GMM-based alignment and parallel SGD with periodic model averaging. `nnet3` is an extension of `nnet2` designed to support a wider variety of networks than simple feed-forward networks (FFNs) [19]. `chain`, based on `nnet3`, uses a 3 times smaller frame rate at the output of the neural network, and uses additional lattice-free maximum mutual information (LF-MMI) for training [20].

In this paper, according to the standard recipe mentioned above, we attempt to improve the two existing acoustic models implemented through the library of `nnet3` without the training criterion of LF-MMI. One is time-delay neural networks (TDNNs), which are also known as one-dimensional convolutional neural networks (1-d CNNs). It has been shown that they can effectively learn the temporal dynamics of signals from

http://dx.doi.org/10.21437/Interspeech.2019-1717

short-term feature representations [21, 22]. The other, built up of interleaving TDNN layers and long short-term memory (LSTM) layers [23], is the TDNN-LSTM structure [24], which has a wide temporal context and performs as well as bidirectional LSTM networks with less latency [25]. Following the work of discriminative autoencoders (DcAEs) for acoustic modeling in [26], where the encoder layers were only implemented by deep FFNs with temporarily augmented fMLLR features, we separately use TDNNs and TDNN-LSTM networks as encoders to see if the advantages of DcAEs can be sustained and developed. As described in [27], an autoencoder can help to preserve the most salient information. Furthermore, as a feature regularizer, it has the advantage of improving generalization of the learned acoustic models by increasing the ability to reconstruct the input features from the frame embeddings. Our work in this paper has at least three major contributions as follows:

1. The structure of our proposed model can be applied effortlessly to most of existing or prospective neural network models without increasing the decoding costs and size of the inference model, demonstrating its flexibility and universality. Based on our modified `nnet3` setup and recipe[3], comparison with other models and application to other corpora can be realized easily in Kaldi.
2. The mechanism of skip connections (or the u-net) is included in DcAEs for better output regularization [28].
3. The results of LVCSR experiments on two corpora, namely the MATBN Mandarin Chinese corpus and the WSJ English corpus, indicate that our proposed method can be applied across languages.

2. Discriminative Autoencoders

For standard ASR tasks built with the Kaldi `nnet3` setup, acoustic features are typically fed into the TDNN or TDNN-LSTM model frame-wisely to extract phoneme-related information for triphone state prediction, as shown in Figure 2. It is worth noting that the actual output vector is a one-hot vector representing a categorical distribution over a set of events, i.e., triphone states. To help the extraction of phoneme-related information, we attached FFN layers as a decoder to the penultimate layer of the TDNN or TDNN-LSTM model and treated the baseline model as an encoder, creating an autoencoder-like structure (cf. DcAE-B in Figure 2).

Autoencoders are learning models designed to convert an input to an output with minimal distortion. The embedded internal representation, namely the code layer, consists of the most critical information that minimizes the reconstruction error. To obtain accurate acoustic scores for ASR, we divided the code layer into two sub-representations: the phoneme-aware layer (p-code) and the residual layer (r-code). The p-code consisted of phonetic information and was connected to the original output layer of an ASR task. The r-code carried information within the acoustic frame that was unrelated to phonetic information, which could include environmental noise, speaker identity, and any other possible information that was considered useless for the task. The p-code and r-code were then concatenated as the input of the decoder for reconstructing the acoustic frame.

On top of the DcAE-B architecture, we further added skip connections between the encoder and the decoder (cf. DcAE-U in Figure 2). Based on the concept of momentum [29], which is a common optimization method that helps speed up SGD in the relevant direction and reduce oscillations by adding a fraction of the update vector from the past time step to the current update vector, we wish to use the information from the encoder as a kind of momentum to guide the training of the autoencoder with skip connections. The implemented structure is similar to the u-net [28], which has achieved great success in medical image segmentation tasks. We connected the outputs of the encoder layer with those of the decoder layer, producing new inputs for the next layer of the decoder as

$$D_{j+1}^{in} = E_i^{out} \oplus \beta D_j^{out}, \tag{1}$$

where E_i^{out} and D_j^{out} denote the outputs of the i^{th} layer of the encoder and j^{th} layer of the decoder, respectively, D_{j+1}^{in} denotes the input of the $j+1^{th}$ layer of the decoder, β is used to control the strength of the momentum, and $\oplus$ denotes the connection operation. In this paper, we use two kinds of skip connections, one by appending and the other by summation.

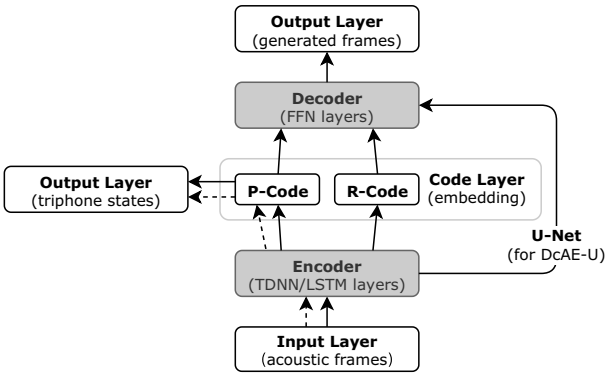

Figure 2: *Architectures of different acoustic modeling systems. The workflow of the baseline, denoted by the dotted directional lines, is composed of the input layer, encoder, p-code, and output layer. Note that the p-code is usually the penultimate layer of the baseline, and the additional FFN layers between the p-code and the output layer, representing the triphone states, are optional. The basic version of DcAEs (DcAE-B), expressed by the solid directional lines, adds the r-code, decoder, and output layer for feature reconstruction as its key components. The u-net based DcAEs (DcAE-U) has additional connections between the encoder and decoder layers.*

2.1. Objective function

The objective function used in the proposed framework includes two parts: the mean square error for feature reconstruction and the phoneme-aware cross-entropy for acoustic modeling.

2.1.1. The reconstruction error

Suppose our proposed model $\mathcal{M}$ contains a pair of deterministic mappings $f(\cdot)$ and $g(\cdot)$, which are responsible for latent variable inference and observation generation in the terminology of Bayesian inference, respectively. Given a set of training data $\mathcal{X}$ ready to go through the inference-generation process $\mathcal{X} \xrightarrow{f} \mathcal{H} \xrightarrow{g} \mathcal{X}$, where $\mathcal{H}$ is the internal representation residing in a latent subspace, i.e., the code layer shown in Figure 2, the average reconstruction error, denoted by $\mathcal{L}_{mse}$, based on the residual sum of squares between $\mathbf{x} \in \mathcal{X}$ and its reconstruction $\mathbf{x}' = g(f(\mathbf{x}))$ is given by

$$\mathcal{L}_{mse}(\mathcal{X}) = \frac{1}{|\mathcal{X}|} \sum_{\mathbf{x} \in \mathcal{X}} \|\mathbf{x}' - \mathbf{x}\|_2^2, \tag{2}$$

[3] https://github.com/melodyhpt/Kaldi-DcAE

where $\|\cdot\|_2^2$ is the 2-norm operator and $|\mathcal{X}|$ is the sample or mini-batch size. Like a copy machine, $\mathcal{H}$ is usually restricted in ways that allow it to copy only approximately, and to copy only input that resembles the training data. Because the model is forced to prioritize which aspects of the input should be copied, it often learns useful properties of the data [27].

2.1.2. The phoneme-aware cross-entropy

One of the straightforward strategies to interpret the phonetic information of an sensory input is to leverage a categorical distribution over the predefined context-dependent phonetic states. In Kaldi, the HMM state of a triphone is assigned with a pdf-id, which is used as an index for the probability distribution function (pdf). These distinct and contiguous pdf-ids are used as targets for acoustic modeling. To realize the representation, we define an objective, expressed by $\mathcal{L}_{xent}$, to maximize the total logarithmic posterior probability of the target pdf-ids over all training samples $\mathcal{X}$ (or minimizing the cross-entropy between the ground truth and the predicted values):

$$\mathcal{L}_{xent}(\mathcal{X}) = -\sum_{\mathbf{x}\in\mathcal{X}} \log p(q_{\mathbf{x}}|\mathbf{x}), \tag{3}$$

where $q_{\mathbf{x}}$ denotes the context-dependent HMM state for $\mathbf{x}$, and $p(q_{\mathbf{x}}|\mathbf{x})$ is obtained through a softmax activation function.

2.1.3. The final objective function

By combining the reconstruction error and the phoneme-aware cross-entropy, the objective function to be minimized becomes

$$(1-\alpha)\mathcal{L}_{xent} + \alpha\mathcal{L}_{mse}, \tag{4}$$

where α is an adjustable weight between $\mathcal{L}_{xent}$ and $\mathcal{L}_{mse}$.

We implemented this framework in Kaldi by modifying its `nnet3` neural network library. We added an additional output layer called `output_ae` as the output layer of the DcAE. The target of this layer was set to the input feature. The weight α was set by adjusting the parameter `learning-rate-factor` of each output layer.

3. Experiments

3.1. Corpora

We evaluated our proposed framework with two speech corpora of different languages, namely Wall Street Journal (WSJ) [30] and Mandarin Chinese Broadcast News (MATBN) [31].

3.1.1. Wall Street Journal (WSJ)

For WSJ, we used both WSJ0 (LDC93S6B) and WSJ1 (LDC94S13B) as the training set, consisting of 81 hours and known as `train_si284` in most Kaldi recipes. Besides, we used `dev93` and `eval92` as the test sets.

3.1.2. Mandarin Chinese Broadcast News (MATBN)

MATBN is a broadcast news corpus collected in Taiwan [31]. Each episode has been segmented into separate stories and manually transcribed. Each story contains the speech of one studio anchor, as well as several field reporters and interviewees. In our experiments, we used a subset of 25-hour speech data for training the model and tested on two testing sets, namely `Dev` and `Test`, each consisting of 1.4 hours of speech.

3.2. Input features

To extract acoustic features, spectral analysis was applied to a 25 ms frame of speech waveform every 10 ms. For each frame, 40 high-resolution MFCCs, derived by DCT conducted on 40 Mel-frequency bins and normalized by utterance-based mean subtraction, were used as the input to the NN-based acoustic models. Since Mandarin is a tonal language, 3 pitch-related features were concatenated to the 40-dimensional MFCCs [32] for the Mandarin ASR task. Moreover, for both tasks, we appended a 100-dimensional i-vector to each acoustic frame [18].

3.3. Baseline systems

The GMM-HMM system was trained to generate frame-to-state (or pdf) alignments for subsequent neural network training. We used the fifth round triphone system (i.e., `tri5`) in MATBN and the fourth round triphone system (i.e., `tri4`) in WSJ to decode and generate the alignment for each utterance.

The TDNN-based system was built with 6 hidden layers, each containing 650 hidden nodes. Their output layer was a softmax-activated layer with 3,376 nodes for WSJ and 4,272 for MATBN, and the maximum change in the parameters per mini-batch was set to 1.5. The detailed splicing indices are given in Table 1. The mini-batch sizes were 256 and 128. The initial and final effective learning rates were set to 0.0015 and 0.00015, respectively, and the total number of training epochs was set to 3, while additional 4 training epochs were used for fine tuning with the state-level minimum Bayes risk (sMBR) criterion [33].

The TDNN-LSTM-based system was constructed by 3 continuous groups of layers, where each group consisted of 2 TDNN-based layers followed by an LSTM-based layer, and a dense layer. Each layer had 520 hidden nodes. We used a delay factor of -3 and decay time of 20 for all LSTM-based layers. The mini-batch sizes were 128 and 64, and the number of training epochs was set to 6. The other hyper-parameters were set to the same values as the TDNN-based system.

3.4. Proposed systems

The dimensions of the residual layer (r-code) and the decoder layers were the same as the number of nodes of the baseline model's penultimate layer (p-code). The number of decoder layers was given as D in the column named Architecture in Table 1. For the proposed u-net version of DcAE (DcAE-U), additional skip-connections were added symmetrically between the second layer and the penultimate layer as well as the third layer and the third to the last layer. In DcAE-U with the appending operation, the dimensions of the decoder layers, which were linked with skip-connections and appended with the corresponding encoder layers, were doubled. Take TDNN-DcAE-U for example, as shown in Table 1, the dimension of the last two dense layers was 1,300 (650×2) for the append version.

3.5. Results

Table 1 presents various models' architectures and layer-wise contexts used in this paper, where DcAE-B represents DcAE without highway connections, and DcAE-U represents DcAE with highway connections. For DcAE-U, both implementation methods (appending and summing) have the same architecture. The layer-wise contexts for the DcAE models were the same as their baseline TDNN or TDNN-LSTM model.

Tables 2 and 3 show the error rates achieved with TDNN and TDNN-LSTM as well as the proposed DcAE-based systems for MATBN and WSJ. For MATBN, α for TDNN was set

Table 1: *Various models' specifications used in WSJ and MATBN. D, T, L, and C denote the dense, TDNN-based, LSTM-based, and code layers, respectively. The subscript of the layer notation means the skip connection between two layers. For example, T_{h1} and D_{h1} have a common connection $h1$. The layer-wise context shows the information about splicing indices of TDNN-based layers, which are usually somewhat different between the TDNN and TDNN-LSTM systems. DcAE-B stands for the basic version of DcAEs while DcAE-U means the version utilizing the u-net.*

Model (for MATBN)	**Architecture**	**Layer-wise context**					
TDNN	$TTTTTD$	{-2,-1,0,1,2}	{-1,0,1}	{-1,0,-1}	{-3,0,3}	{-6,-3,0}	
TDNN-DcAE-B	$TTTTTCDDDDD$						
TDNN-DcAE-U	$T_{h1}T_{h2}TTTCDDDD_{h2}D_{h1}$						
TDNN-LSTM	$TTLTTLTTLD$	{-2,-1,0,1,2}	{-1,0,1}	{-3,0,3}	{-3,0,3}	{-3,0,3}	{-3,0,3}
TDNN-LSTM-DcAE-B	$TTLTTLTTLCDD$						
TDNN-LSTM-DcAE-U	$T_{h1}T_{h2}LTTLTTLCD_{h2}D_{h1}$						
Model (for WSJ)	**Architecture**	**Layer-wise context**					
TDNN	$TTTTTD$	{-2,-1,0,1,2}	{-1,0,1}	{-1,0,-1}	{-3,0,3}	{-6,-3,0}	
TDNN-DcAE-B	$TTTTTCDDD$						
TDNN-DcAE-U	$T_{h1}T_{h2}TTTCDD_{h2}D_{h1}$						
TDNN-LSTM	$TTLTTLTTLD$	{-2,-1,0,1,2}	{-1,0,1}	{-3,0,3}	{-3,0,3}	{-3,0,3}	{-3,0,3}
TDNN-LSTM-DcAE-B	$TTLTTLTTLCDDDD$						
TDNN-LSTM-DcAE-U	$T_{h1}T_{h2}LTTLTTLCDDD_{h2}D_{h1}$						

Table 2: *Character error rates (%) with respect to two baselines and their relevant DcAE-based variants for MATBN.*

Model	**Dev**	**Test**	**+sMBR Dev**	**+sMBR Test**
TDNN	7.88	7.64	7.54	7.45
DcAE-B	7.45	**7.39**	7.20	**7.07**
DcAE-U (append)	**7.43**	**7.39**	7.21	7.22
DcAE-U (sum)	7.51	**7.39**	**7.17**	7.12
TDNN-LSTM	8.37	8.26	8.05	7.92
DcAE-B	8.05	7.92	7.64	7.68
DcAE-U (append)	**7.78**	7.91	**7.28**	**7.37**
DcAE-U (sum)	8.00	**7.86**	7.72	7.62

Table 3: *Word error rates (%) with respect to two baselines and their relevant DcAE-based variants for WSJ.*

Model	**dev93**	**eval92**	**+sMBR dev93**	**+sMBR eval92**
TDNN	6.62	3.92	5.93	3.51
DcAE-B	**6.23**	**3.58**	6.07	**3.23**
DcAE-U (append)	6.52	3.79	5.94	3.46
DcAE-U (sum)	6.36	3.67	**5.88**	3.44
TDNN-LSTM	6.29	4.00	6.39	3.99
DcAE-B	6.45	3.90	6.23	**3.62**
DcAE-U (append)	6.56	4.09	6.25	3.92
DcAE-U (sum)	**6.21**	**3.77**	**6.01**	3.7

to 5×10^{-10} and β was set to 10^{-1}; α for TDNN-LSTM was set to 5×10^{-9} and β was set to 8×10^{-1}. Note that in actual implementation, $\mathcal{L}_{mse}$ in Eq. (4) is not an average as shown in Eq. (2) but is orders of magnitude larger than $\mathcal{L}_{xent}$, so it is not surprising that α is a very small value. Compared to the TDNN baseline system, DcAE-B before and after fine-tuning with sMBR achieved relative error rate reductions of 3% and 5%, respectively. With the TDNN-LSTM-based architecture, DcAE-U achieved a relative error reduction of 4% over DcAE-B both before and after fine-tuning with sMBR. For WSJ, α for TDNN was set to 10^{-10} and β was set to 5×10^{-1}; α for TDNN-LSTM was set to 10^{-10} and β was set to 3×10^{-1}. The DcAE-based systems generally outperformed the baseline models with relative error rate reductions of 5% to 10%.

From the results, we can conclude that DcAE-B in general outperforms its baseline model with relative error rate reductions of 5% to 10% for both corpora. The results show that the reconstruction error $\mathcal{L}_{mse}$ can indeed help extract the most salient information, making it easier to separate out the phonetic information. The results also indicate that, despite having a great success in the field of image recognition, adding skip-connections does not necessarily benefit ASR. A possible reason is due to the different types of information to be learned by different systems. For image recognition tasks, the knowledge learned by the model is space-wise information extracted from an image. However, for ASR tasks, the goal is to learn time-wise information, which is sequential.

4. Conclusions and Future Work

This paper presented two kinds of discriminative autoencoders (DcAEs), namely DcAE-B and DcAE-U, which were implemented on two existing acoustic models, namely TDNN and TDNN-LSTM, on the basis of the `nnet3` setup of Kaldi. The results showed that DcAE-based systems achieved error rate reductions over their baseline systems for both Mandarin and English ASR tasks. We also provided the DcAE recipe for WSJ, allowing rapid reproduction of our results. We will implement DcAEs on other outstanding models such as TDNN-F [34] and CNN-DNN [35] as well as other larger data sets. In addition, inspired by the concept of sequence to sequence leaning [11] and deconvolutional networks [36], the relationship of adjacent phonetic embeddings (p-code) will be taken into account for the design of the decoder in DcAEs. We will also implement DcAE-based models on the Kaldi `chain` setup [37].

Acknowledgements: This work was supported in part by the MOST-Taiwan Grants 105-2221-E-001-012-MY3 and 108-2634-F-001-004.

5. References

[1] G. Hinton, L. Deng, D. Yu, G. Dahl, A. R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[2] L. Deng and X. Li, "Machine Learning Paradigms for Speech Recognition : An Overview," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 5, pp. 1060–1089, 2013.

[3] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlíček, Y. Qian, P. Schwarz, J. S. Silovsky, G. Stemmer, and K. V. Vesely, "The Kaldi Speech Recognition Toolkit," in *Proc. IEEE ASRU*, 2011.

[4] A. Graves, S. Fernandez, F. Gomez, and J. Schmidhuber, "Connectionist Temporal Classification : Labelling Unsegmented Sequence Data with Recurrent Neural Networks," *Proc. ICML*, 2006.

[5] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, and A. Y. Ng, "Deep Speech: Scaling up End-to-end Speech Recognition," 2014. [Online]. Available: https://arxiv.org/abs/1412.5567

[6] A. Senior, H. Sak, F. d. C. Quitry, T. Sainath, and K. Rao, "Acoustic Modelling with CD-CTC-SMBR LSTM RNNS," in *Proc. IEEE ASRU*, 2015.

[7] H. Soltau, H. Liao, and H. Sak, "Neural Speech Recognizer : Acoustic-to-Word LSTM Model for Large Vocabulary Speech Recognition," in *Proc. Interspeech*, 2017.

[8] K. Audhkhasi, B. Ramabhadran, G. Saon, M. Picheny, and D. Nahamoo, "Direct Acoustics-to-Word Models for English Conversational Speech Recognition," in *Proc. Interspeech*, 2017.

[9] J. Li, G. Ye, R. Zhao, J. Droppo, and Y. Gong, "Acoustic-to-Word Model without OOV," in *Proc. IEEE ASRU*, 2017.

[10] J. Li, G. Ye, A. Das, R. Zhao, and Y. Gong, "Advancing Acoustic-to-Word CTC Model," in *Proc. IEEE ICASSP*, 2018.

[11] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to Sequence Learning with Neural Networks," in *Proc. NIPS*, 2014.

[12] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," in *Proc. ICLR*, 2015.

[13] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, "End-to-end Continuous Speech Recognition Using Attention-based Recurrent NN: First Results," in *Proc. Deep Learning and Representation Learning Workshop, NIPS*, 2014.

[14] A. Das, J. Li, R. Zhao, and Y. Gong, "Advancing Connectionist Temporal Classification with Attention," in *Proc. IEEE ICASSP*, 2018.

[15] T. Ko, V. Peddinti, D. Povey, S. Khudanpur, H. Noah, and H. Kong, "Audio Augmentation for Speech Recognition," in *Proc. Interspeech*, 2015.

[16] D. Snyder, G. Chen, and D. Povey, "MUSAN: A Music, Speech, and Noise Corpus," 2015. [Online]. Available: https://arxiv.org/abs/1510.08484

[17] T. Ko, V. Peddinti, D. Povey, M. L. Seltzer, and S. Khudanpur, "A Study on Data Augmentation of Reverberant Speech for Robust Speech Recognition," in *Proc. IEEE ICASSP*, 2017.

[18] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end Factor Analysis for Speaker Verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[19] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-Dependent Pre-Trained Deep Neural Networks for Large-Vocabulary Speech Recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 30–42, 2012.

[20] V. Manohar, H. Hadian, D. Povey, and S. Khudanpur, "Semi-supervised Training of Acoustic Models Using Lattice-free MMI," in *Proc. IEEE ICASSP*, 2018.

[21] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, 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.

[22] V. Peddinti, D. Povey, and S. Khudanpur, "A Time Delay Neural Network Architecture for Efficient Modeling of Long Temporal Contexts," in *Proc. Interspeech*, 2015.

[23] S. Hochreiter and S. Jurgen, "Long Short-term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[24] V. Peddinti, Y. Wang, D. Povey, and S. Khudanpur, "Low Latency Acoustic Modeling Using Temporal Convolution and LSTMs," *IEEE Signal Processing Letters*, vol. 25, no. 3, pp. 373–377, 2018.

[25] H. Hadian, H. Sameti, D. Povey, and S. Khudanpur, "End-to-end Speech Recognition Using Lattice-free MMI," in *Proc. Interspeech*, 2018.

[26] M. H. Yang, H. S. Lee, Y. D. Lu, K. Y. Chen, Y. Tsao, B. Chen, and H. M. Wang, "Discriminative Autoencoders for Acoustic Modeling," in *Proc. Interspeech*, 2017.

[27] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. The MIT Press, 2016.

[28] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional Networks for Biomedical Image Segmentation," in *Proc. MICCAI*, 2015.

[29] N. Qian, "On the Momentum Term in Gradient Descent Learning Algorithms," *Neural Networks*, vol. 12, no. 1, pp. 145–151, 1999.

[30] D. B. Paul and J. M. Baker, "The Design for the Wall Street Journal-based CSR Corpus," in *Proc. DARPA Speech and Natural Language Workshop*, 1992.

[31] H.-M. Wang, B. Chen, J.-W. Kuo, and S.-S. Cheng, "MATBN: A Mandarin Chinese Broadcast News Corpus," *International Journal of Computational Linguistics & Chinese Language Processing*, vol. 10, no. 2, pp. 219–236, 2005.

[32] P. Ghahremani, B. Babaali, D. Povey, K. Riedhammer, J. Trmal, and S. Khudanpur, "A Pitch Extraction Algorithm Tuned for Automatic Speech Recognition," in *Proc. IEEE ICASSP*, 2014.

[33] K. Vesely, A. Ghoshal, L. Burget, and D. Povey, "Sequence-discriminative Training of Deep Neural Networks," in *Proc. Interspeech*, 2013.

[34] D. Povey, G. Cheng, Y. Wang, K. Li, H. Xu, M. Yarmohamadi, and S. Khudanpur, "Semi-Orthogonal Low-Rank Matrix Factorization for Deep Neural Networks," in *Proc. Interspeech*, 2018.

[35] T. N. Sainath, A.-R. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep Convolutional Neural Networks for LVCSR," in *Proc. IEEE ICASSP*, 2013.

[36] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, "Deconvolutional Networks," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2528–2535, 2010.

[37] D. Povey, V. Peddinti, D. Galvez, P. Ghahrmani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, "Purely Sequence-trained Neural Networks for ASR Based on Lattice-free MMI," in *Proc. Interspeech*, 2016.