*INTERSPEECH 2017*
August 20–24, 2017, Stockholm, Sweden

# Discriminative Autoencoders for Acoustic Modeling

*Ming-Han Yang*[1], *Hung-Shin Lee*[1], *Yu-Ding Lu*[2], *Kuan-Yu Chen*[1],
*Yu Tsao*[2], *Berlin Chen*[3], *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 Normal University, Taiwan

{mhyang, hslee, kychen, whm}@iis.sinica.edu.tw, berlin@ntnu.edu.tw

## Abstract

Speech data typically contain information irrelevant to automatic speech recognition (ASR), such as speaker variability and channel/environmental noise, lurking deep within acoustic features. Such unwanted information is always mixed together to stunt the development of an ASR system. In this paper, we propose a new framework based on autoencoders for acoustic modeling in ASR. Unlike other variants of autoencoder neural networks, our framework is able to isolate phonetic components from a speech utterance by simultaneously taking two kinds of objectives into consideration. The first one relates to the minimization of reconstruction errors and benefits to learn most salient and useful properties of the data. The second one functions in the middlemost code layer, where the categorical distribution of the context-dependent phone states is estimated for phoneme discrimination and the derivation of acoustic scores, the proximity relationship among utterances spoken by the same speaker are preserved, and the intra-utterance noise is modeled and abstracted away. We describe the implementation of the discriminative autoencoders for training tri-phone acoustic models and present TIMIT phone recognition results, which demonstrate that our proposed method outperforms the conventional DNN-based approach.

**Index Terms**: acoustic modeling, automatic speech recognition, discriminative autoencoders, deep neural networks

## 1. Introduction

Recently, significant improvements in speech recognition accuracy have been obtained in a variety of tasks whose acoustic models were based on deep neural networks (DNN) rather than Gaussian mixture models (GMM) [1, 2]. More recent advances in deep learning techniques applied to speech processing include the locally-connected or convolutional deep neural networks (CNN) [3, 4, 5], temporally recurrent versions of neural networks (RNN) [6, 7], or the combination of the previous models [8]. Besides, another branch of research focuses on how to directly and independently employ neural networks with no reliance on hidden Markov models (HMM) for sequence modeling. For example, the model based on long-short term memory (LSTM) took a successful step for the replacement of the standard DNN-HMM hybrid framework [9, 10, 11].

All of the above methods are related to discriminative modeling. However, building rich generative models, which are capable of extracting or preserving useful, salient, and high-level latent information from the high-dimensional, context-embedded sensory input, lies at the core of solving many machine learning tasks [12]. These models capture the underlying structure of data by defining flexible probability distributions over high-dimensional data as part of a complex, partially observed system. Recently, some of the successful generative models, such as variational autoencoders (VAE) [13, 14] and generative adversarial networks (GAN) [15], were proposed to discover meaningful and high-level latent representations. In light of these developments, we attempt to propose a new framework, named discriminative autoencoders (DcAE), for acoustic modeling based on autoencoder neural networks. With a proper use of sample labels, not only can the optimization of DcAEs be handled in a discriminative fashion, but the whole model inherits the aforementioned merits of generative models.

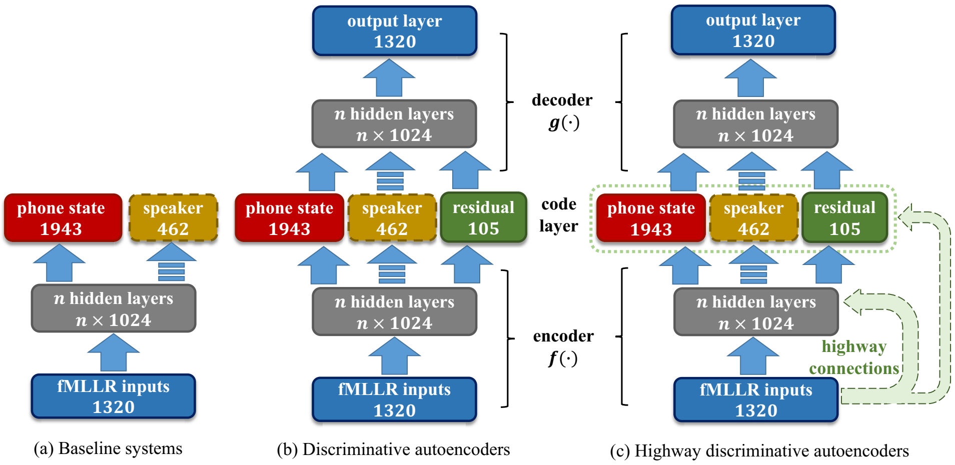
We replace the role of DNN with an autoencoder and tweak its objectives for phoneme classification and speaker discrimination. The autoencoder is a symmetric neural network that is trained to approximately copy its input to the output [16]. In addition to the reconstruction error, which makes the autoencoder analogous to a generative model that benefits to unsupervisedly learn most salient and useful properties of the data, three additional objective functions are also considered in our proposed framework. The first one is used to estimate the categorical distribution, i.e., the posterior probabilities of the context-dependent phone states, in the middlemost code layer to form a phoneme code vector (p-vector) for each acoustic frame. The other two objectives attempt to ensure that utterances spoken by the same speaker would have similar representations in the speaker-discriminative subspace represented in the code layer. Finally, the acoustic score of each acoustic frame is derived by its p-vector. To our best knowledge, although autoencoders have been widely applied to many speech processing tasks, such as speech enhancement [17, 18], acoustic novelty detection [19], and robust feature extraction in ASR [20, 21, 22], fewer papers used them directly for acoustic modeling.

Our work in this paper continues the general line of DcAE research in [23] and has at least the following two major contributions. First, our proposed model is able to isolate phonetic components from a speech utterance by simultaneously taking two kinds of objectives into consideration. Second, along the same vein, we also apply additional highway connections between inputs, hidden layers, and code layers, showing the flexibility of the proposed DcAE structure.

## 2. Discriminative Autoencoders and Objective Functions

Formally, given a set of training data $\mathcal{X}$, a classic autoencoder is used to infer internal representations or codes $\mathcal{H}$ by minimizing the reconstruction error between each input $\mathbf{x} \in \mathcal{X}$ and its reconstructed output $\mathbf{x}'$. Thus, we can suppose $\mathcal{H}$ contains untreated phonetic, speaker, and other characteristics [16]. In order to extract the phonetic information so as to estimate accurate

http://dx.doi.org/10.21437/Interspeech.2017-221



(a) Baseline systems (b) Discriminative autoencoders (c) Highway discriminative autoencoders

Figure 1: *Illustrations of different acoustic modeling systems. The numbers in each block denote the number of nodes in the corresponding layer. (a) Two baseline systems, DNNs and Multi-tasking DNNs. (b) DcAEs with or without the speaker identity layer, expressed by a dotted block. (c) Highway DcAEs, where the green arrows express the highway connections from the input layer to each hidden layer as well as the middlemost code layer.*

acoustic scores for ASR, we assume that the internal representation can be separated into three sub-components: the phoneme-aware vector (p-code), the speaker identity vector (s-code), and the residual vector (noise within the acoustic frame). In other words, the internal representation of a given acoustic sample is concatenated by three individual vector representations. In the following subsections, we present three kinds of objective functions to be used for training an autoencoder.

## 2.1. 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, the average reconstruction error 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

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

where $\|\cdot\|_2^2$ is the 2-norm operator and $|\mathcal{X}|$ is the sample 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.

## 2.2. 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. To realize the representation, we define an objective function that aims at maximizing the log posterior probability over all training samples $\mathcal{X}$ (or minimizing the cross-entropy between the ground truth and the predicted values):

$$\mathcal{L}_{phn-ce}(\mathcal{X}) = \sum_{\mathbf{x} \in \mathcal{X}} \log p(q_{\mathbf{x}}|\mathbf{x}), \tag{2}$$

where $q_{\mathbf{x}}$ denotes the context-dependent phone state for $\mathbf{x}$, and $p(q_{\mathbf{x}}|\mathbf{x})$ is obtained through a softmax activation function. It is worthy to note that the dimension of p-code is equivalent to the number of distinct context-dependent phonetic states.

## 2.3. The speaker-aware loss

The speaker-aware loss function $\mathcal{L}_{spk}$, acting in the speaker identity code (s-code) layer, can be realized in two alternative ways: 1) the cross-entropy loss and 2) the within-speaker scatterness along with the between-speaker ambiguity.

### 2.3.1. *The speaker-aware cross-entropy*

For the purpose of segregating the speaker statistics from others, a simple but direct approach is to first represent a speaker by a one-hot representation, where each element corresponds to a distinct speaker in the training set. Then, the training objective can be defined by maximizing the probability of the true speaker given an acoustic observation $\mathbf{x}$:

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

where $s_{\mathbf{x}}$ denotes the speaker of $\mathbf{x}$, and $p(s_{\mathbf{x}}|\mathbf{x})$ is computed by feedforward from the input sample to the speaker identity code layer with a softmax activation function. The deduced probability distribution is treated as the s-code.

### 2.3.2. *The within-speaker scatterness*

Suppose $\mathcal{H}_{spk} = \{\mathcal{H}_{s1}, ..., \mathcal{H}_{sm}\}$, where $\mathcal{H}_{si}$ corresponds to the training data that belong to the $i$-th speaker of the $m$ training speakers, and the loss function that measures the average scatterness within each speaker is described as follows:

$$\mathcal{L}_{spk-ws}(\mathcal{H}_{spk}) = \frac{1}{m} \sum_{i=1}^{m} \sum_{\mathbf{h} \in \mathcal{H}_{si}} \|\mathbf{h} - \bar{\mathbf{h}}_{si}\|_2^2, \tag{4}$$

where $\bar{\mathbf{h}}_{si}$ denotes the empirical mean vector of $\mathcal{H}_{si}$. Apparently, (4) measures the average within-speaker compactness on the negative side. Therefore, minimizing (4) implies increasing the similarity score between two utterances of the same speaker if the score metric is 2-norm-related. In this way, false rejections in a speaker verification task will be reduced to some extent.

#### *2.3.3. The between-speaker ambiguity*

The last objective function to be minimized is about the ambiguity between s-codes that belong to different speakers, which can be measured by the following loss function:

$$\mathcal{L}_{spk-ba}(\mathcal{H}_{spk}) = -\frac{1}{m}\sum_{i=1}^{m} |\mathcal{H}_{si}| \|\bar{\mathbf{h}}_{si} - \bar{\mathbf{h}}_{s}\|_2^2, \qquad (5)$$

where $\bar{\mathbf{h}}_s$ denotes the empirical mean vector of $\mathcal{H}_{spk}$, and $|\mathcal{H}_{si}|$ stands for the size of $\mathcal{H}_{si}$. It can be seen that larger $\mathcal{L}_{spk-ba}$ occurs when each $\bar{\mathbf{h}}_{si}$ is drawing closer to $\bar{\mathbf{h}}_s$, which makes s-codes more ambiguous in speaker discrimination.

### 2.4. The proposed DcAE framework

Finally, by combining the reconstruction error in (1), the phoneme-aware cross entropy in (2) and the speaker-aware loss in either (3), or (4) and (5), our goal is to find a set of model parameters $\mathcal{M}$ by minimizing

$$\mathcal{L}_{phn-ce} + \alpha\mathcal{L}_{spk} + \mathcal{L}_r + \lambda\|\mathcal{M}\|_2^2, \qquad (6)$$

where $\alpha > 0$ is used to control the degree of reliance on the speaker-aware objective function $\mathcal{L}_{spk}$, $\lambda$ is a regularization parameter that controls the model complexity, and

$$\mathcal{L}_{spk} = \begin{cases} \mathcal{L}_{spk-ce}(\mathcal{X}), \text{ or} \\ \mathcal{L}_{spk-ws}(\mathcal{H}_{spk}) + \mathcal{L}_{spk-ba}(\mathcal{H}_{spk}). \end{cases} \qquad (7)$$

The treatment of $\alpha$ and $\lambda$ is akin to that in ElasticNet [26].

One of the major reasons for using the combination of the three kinds of loss functions is to simultaneously isolate the phonetic and speaker factors that explain the input acoustic features. First, $\mathcal{L}_r$ guarantees that the code layer of DcAEs possesses enough acoustic information to restore or *generate* the observations, just like a generative model based on factor analysis, so that we do not need to worry that the information for phone discrimination will be washed away even if the mismatch between training and test data gets more serious. Second, $\mathcal{L}_{spk}$ and $\mathcal{L}_{phn-ce}$ play a role like a filter to keep what is useful and throw what is worthless into the residual part of the code layer. (This also explains why the residual part is necessary in the code layer.) Therefore, they help centralize the phonetic and speaker information, respectively, into their corresponding identity parts of the code layer, in order to form an s-code and a p-code for each acoustic frame, where the p-code just can be used as its acoustic scores for the subsequent phone decoding.

## 3. Experiments

### 3.1. Experiment setup

Experiments were conducted on the TIMIT corpus [27]. The training set contained 3,696 sentences from 462 speakers. A separate development set of 50 speakers was used for hyperparameter tuning. Experiments were evaluated on the core test set, which consisted of 192 utterances, with 8 utterances from each of the 24 speakers. The speakers in the training, development, and core test sets do not overlap.

The phone recognizer was implemented with three-state triphone HMMs, which were tied to give 1,943 context-dependent phonetic states in total. The likelihoods produced by the DNN were used as the emission probabilities of HMM states. A bigram language model estimated from the phone transcriptions available in the training set was used in the decoding phase. After decoding, the 48 phone labels were mapped to 39 phone classes for scoring, following [28].

### 3.2. Baseline systems

The baseline DNN-HMM system was trained with 2 hidden layers, with 1,024 hidden units in each layer. The input layer consisted of a context window of 11 frames of acoustic features, including the target frame at time $t$, 5 preceding and 5 following frames. Each frame was represented by 40 dimensional feature space maximum likelihood linear regression (fMLLR) features [29], plus their first and second order derivatives. Therefore, each frame was represented by a 120-dimensional vector, and the input layer consisted of 1,320 nodes. The output layer of the baseline system was a softmax layer, with one output node for each of the 1,943 acoustic states. The training target of the DNN was a one-hot representation for each frame. The labels were determined by Viterbi alignment on the training data according to a baseline GMM-HMM recognizer.

The DNN was fine-tuned using back-propagation over the full structure with the framewise cross-entropy criterion. The learning rate was first set to 0.01 for a minimum of 4 epoches. When the validation loss reduction between successive epoches was less than 0.002, the learning rate was halved, and continued to be halved after each epoch for 10 times till the training process stopped. The mini-batch size was 256.

The second baseline system used a multi-task learning DNN (MTL-DNN), which had the same network architecture as the DNN used in the DNN-HMM baseline system, except the output layer. In addition to the output layer with a softmax activation function for the primary task of context-dependent phonetic state classification, the final layer was augmented with a secondary softmax-activated layer for an auxiliary speaker classification task. The MTL-DNN adopted the same learning process as the DNN. The likelihoods produced by the primary task were used as the emission probabilities of HMM states, and the likelihoods produced by the secondary task were discarded.

The GMM-HMM system and the decoders for all the neural network-based systems were built up on top of the Kaldi toolkit [30]. Neural network training was performed on NVIDIA GeForce GTX TITAN X GPUs using an in-house tool based on the Keras library [31].

### 3.3. Results

Table 1 summarizes the neural network models that are compared in the paper and their objective functions, where "MTL–DNN" stands for the multi-task learning DNN, which uses speaker identification as the auxiliary task; "DcAE–1" represents the DcAE model trained by considering the reconstruction error ($\mathcal{L}_r$) and phoneme-aware cross-entropy loss ($\mathcal{L}_{phn-ce}$); "DcAE–2" is trained with $\mathcal{L}_r$, $\mathcal{L}_{phn-ce}$ and the speaker-aware cross-entropy ($\mathcal{L}_{spk-ce}$); "DcAE–3" denotes the model trained with $\mathcal{L}_{\text{r}}$, $\mathcal{L}_{phn-ce}$, the within-speaker scatterness ($\mathcal{L}_{spk-ws}$), and the between-speaker ambiguity ($\mathcal{L}_{spk-ba}$); and "H–DcAE" is an extension of "DcAE–3" by adding extra highway connections in the encoder part in order to ensure that the information from the input layer is kept when training the encoder. For the DcAE-based models, there were two hidden layers in the encoder and decoder parts, respectively. Each hidden layer contained 1,024 nodes. Initial weights were uniformly sampled

Table 1: *The loss functions used in different models.*

| Models | Loss Functions |
|---|---|
| MTL–DNN | $\mathcal{L}_{phn-ce}$, $\mathcal{L}_{spk-ce}$ |
| DcAE–1 | $\mathcal{L}_r$, $\mathcal{L}_{phn-ce}$ |
| DcAE–2 | $\mathcal{L}_r$, $\mathcal{L}_{phn-ce}$, $\mathcal{L}_{spk-ce}$ |
| DcAE–3 | $\mathcal{L}_r$, $\mathcal{L}_{phn-ce}$, $\mathcal{L}_{spk-ws}$, $\mathcal{L}_{spk-ba}$ |
| H–DcAE | $\mathcal{L}_r$, $\mathcal{L}_{phn-ce}$, $\mathcal{L}_{spk-ws}$, $\mathcal{L}_{spk-ba}$ |

Table 2: *Phone error rates (%) by different ASR systems.*

| Models | Loss Weights | PER (Dev) | PER (Test) |
|---|---|---|---|
| DNN | 1 | 21.40 | 20.50 |
| MTL–DNN | 1:0.1 | 21.50 | 20.20 |
| DcAE–1 | 1:1 | 21.20 | 20.70 |
| DcAE–2 | 1:1:0.1 | 21.20 | 20.00 |
| DcAE–3 | 1:1:0.5:0.5 | 21.40 | 19.90 |
| H–DcAE | 1:1:1:1 | 20.10 | **19.50** |

Table 3: *Phone error rates (%) by the pretrained DcAE-based systems.*

| Models | Loss Weights | PER (Dev) | PER (Test) |
|---|---|---|---|
| PT–DcAE–1 | 1:1 | 20.90 | 19.70 |
| PT–DcAE–2 | 1:1:0.1 | 20.90 | 19.80 |
| PT–DcAE–3 | 1:1:0.5:0.5 | 20.90 | **19.60** |
| PT–H–DcAE | 1:1:1:1 | 20.90 | 20.00 |

by the Glorot process, which was fit for the $tanh$ activation function [32]. The adaptive gradient algorithm for updating the model parameters was AdaGrad [33].

Figure 2 shows the phone error rates (PERs) achieved by different DcAE-based models with different numbers of nodes in the residual layer. The speaker loss weight $\alpha$ in (6a) was set to 1.0. The result indicates that increasing the residual layer size does not reduce the PERs. Figure 3 further shows the PERs achieved by "DcAE–2", "DcAE–3" and "H–DcAE" with different $\alpha$ in (6a) and different numbers of nodes in the residual layer. The result shows that "DcAE–2" with a small $\alpha$ in general gives lower PERs. In contrast, "DcAE–3" seems to be more robust against the variation of the speaker loss weight than "DcAE–2", and "H–DcAE" performs well at a larger $\alpha$. The larger $\alpha$ also indicates that the hybrid speaker loss in "DcAE–3" and "H–DcAE" might be more effective than the speaker loss in "DcAE–2", and thus can contribute more in the training objective.

The PERs by different ASR systems are summarized in Table 2. All the DcAE-based systems were implemented with 105 nodes in the residual layer. The loss weights for the loss functions in Table 1 were set according to the development set. It is clear that "DcAE–2", "DcAE–3", and "H–DcAE" outperform "DNN", "MTL–DNN", and "DcAE–1". The result indicates that considering the speaker-aware loss (either the speaker-aware cross-entropy ($\mathcal{L}_{spk-ce}$) in "DcAE–2" or the within-speaker scatterness ($\mathcal{L}_{spk-ws}$) plus the between-speaker ambiguity ($\mathcal{L}_{spk-ba}$) in both "DcAE–3" and "H–DcAE") in the training objective can indeed reduce the PER. Moreover, the combination of $\mathcal{L}_{spk-ws}$ and $\mathcal{L}_{spk-ba}$ are more effective than $\mathcal{L}_{spk-ce}$. "H–DcAE" has the lowest PER of 19.50%, as expected.

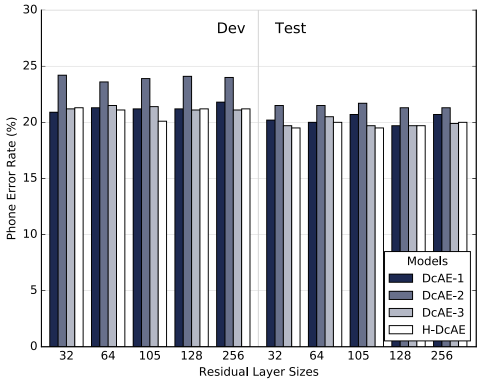


Figure 2: *Phone error rates by different DcAE-based systems with different residual layer sizes.*

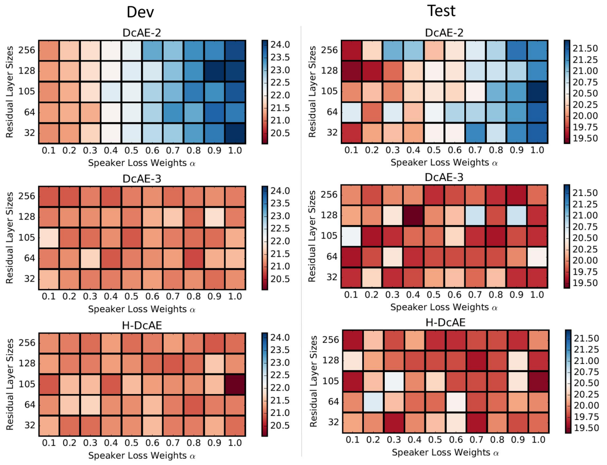


Figure 3: *Heat map in PERs generated by different DcAE-based systems with different $\alpha$ and residual layer sizes.*

### 3.4. Pretraining DcAEs

Gradient-based optimization methods can result in a poor local optimum. To remedy this problem, pretraining methods have been proposed to initialize the parameters prior to back propagation. In some sense, this is similar to using maximum likelihood acoustic models as the initialization for discriminative training in traditional GMM-HMM acoustic models. We pretrained the DcAE model with the reconstruction loss only, and used the model with the lowest validation loss as the initialization for further training using the proposed losses in Secion 2.1, 2.2 and 2.3. From the results shown in Tables 2 and 3, we find that pretraining can improve the performance of DcAE models, in most cases.

## 4. Conclusions

In this paper, we have proposed a new framework based on autoencoders for acoustic modeling in ASR. Unlike other variants of autoencoder neural networks, our framework is able to isolate phonetic components from a speech utterance by simultaneously taking two kinds of objectives into consideration, which cooperate to make our model behave like a generative model that possesses discriminative power for ASR. The experimental results demonstrate that our proposed framework outperforms the conventional DNN-based methods.

## 5. Acknowledgements

This work was supported in part by the Ministry of Science and Technology of Taiwan under Grant: MOST 105-2221-E-001-012-MY3.

# 6. References

[1] T. N. Sainath, B. Kingsbury, B. Ramabhadran, P. Fousek, P. Novak, and A.-R. Mohamed, "Making deep belief networks effective for large vocabulary continuous speech recognition," *in Proc. ASRU*, 2011.

[2] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. 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.

[3] L. Tth, "Modeling long temporal contexts in convolutional neural network-based phone recognition," *in Proc. ICASSP*, 2015.

[4] Y. Hoshen, R. J. Weiss, and K. W. Wilson, "Speech acoustic modeling from raw multichannel waveforms," *in Proc. ICASSP*, 2015.

[5] T. N. Sainath, A.-R. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for LVCSR," *in Proc. ICASSP*, 2013.

[6] T. N. Sainath and B. Li, "Modeling time-frequency patterns with lstm vs. convolutional architectures for LVCSR tasks," *in Proc. Interspeech*, 2016.

[7] Y. Zhang, D. Yu, M. L. Seltzer, and J. Droppo, "Speech recognition with prediction-adaptation-correction recurrent neural networks," *in Proc. ICASSP*, 2015.

[8] T. N. Sainath, A. W. Senior, O. Vinyals, and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks," *in Proc. ICASSP*, 2015.

[9] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks." *in Proc. ICASSP*, 2013.

[10] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks." *in Proc. ICML*, 2014.

[11] L. Lu, X. Zhang, and S. Renais, "On training the recurrent neural network encoder-decoder for large vocabulary end-to-end speech recognition," *in Proc. ICASSP*, 2016.

[12] Y. Tang, N. Srivastava, and R. R. Salakhutdinov, "Learning generative models with visual attention," *in Proc. NIPS*, 2014.

[13] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *in Proc. ICLR*, 2014.

[14] S. Tan and K. C. Sim, "Learning utterance-level normalisation using variational autoencoders for robust automatic speech recognition," *in Proc. SLT*, 2016.

[15] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *in Proc. NIPS*, 2014.

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

[17] S. Araki, T. Hayashi, M. Delcroix, M. Fujimoto, K. Takeda, and T. Nakatani, "Exploring multi-channel features for denoising-autoencoder-based speech enhancement," *in Proc. ICASSP*, 2015.

[18] O. Plchot, L. Burget, H. Aronowitz, and P. Matjka, "Audio enhancing with DNN autoencoder for speaker recognition," *in Proc. ICASSP*, 2016.

[19] E. Marchi, F. Vesperini, F. Eyben, S. Squartini, and B. Schuller, "A novel approach for automatic acoustic novelty detection using a denoising autoencoder with bidirectional lstm neural networks," *in Proc. ICASSP*, 2015.

[20] J. Gehring, Y. Miao, F. Metze, and A. Waibel, "Extracting deep bottleneck features using stacked auto-encoders," *in Proc. ICASSP*, 2013.

[21] X. Zheng, Z. Wu, H. Meng, and L. Cai, "Contrastive auto-encoder for phoneme recognition," *in Proc. ICASSP*, 2014.

[22] T. N. Sainath, B. Kingsbury, and B. Ramabhadran, "Auto-encoder bottleneck features using deep belief networks," *in Proc. ICASSP*, 2012.

[23] H.-S. Lee, Y.-D. Lu, C.-C. Hsu, Y. Tsao, H.-M. Wang, and S.-K. Jeng, "Discriminative autoencoders for speaker verifcation." *in Proc. ICASSP*, 2017.

[24] K. Fukunaga, *Introduction to statistical pattern recognition*. Academic press, 2013.

[25] H.-S. Lee and B. Chen, "Linear discriminant feature extraction using weighted classification confusion information." *in Proc. Interspeech*, 2008.

[26] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 67, no. 2, pp. 301–320, 2005.

[27] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, and D. S. Pallett, "Darpa TIMIT acoustic-phonetic continuous speech corpus CD-ROM. NIST speech disc 1-1.1," *NASA STI/Recon technical report*, vol. 93, 1993.

[28] K. F. Lee and H. W. Hon, "Speaker-independent phone recognition using hidden Markov models," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 37, no. 11, pp. 1641–1648, 1989.

[29] M. J. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Computer Speech & Language*, vol. 12, no. 2, pp. 75–98, 1998.

[30] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The Kaldi speech recognition toolkit," *in Proc. ASRU*, 2011.

[31] F. Chollet, "Keras: Deep learning library for theano and tensorflow," https://github.com/fchollet/keras, 2015.

[32] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks." *in Proc. AISTATS*, 2010.

[33] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.