Interspeech 2018
2-6 September 2018, Hyderabad

Training Utterance-level Embedding Networks for Speaker Identification and Verification

Heewoong Park[1], *Sukhyun Cho*[1], *Kyubyong Park*[2], *Namju Kim*[2], *Jonghun Park*[1]

[1]Dept. of Industrial Engineering & Center for Superintelligence, Seoul National University, Republic of Korea
[2]Kakao Brain Corporation, Republic of Korea

hee188@snu.ac.kr, chosh90@snu.ac.kr, kyubyong.park@kakaobrain.com, namju.kim@kakaobrain.com, jonghun@snu.ac.kr

Abstract

Encoding speaker-specific characteristics from speech signals into fixed length vectors is a key component of speaker identification and verification systems. This paper presents a deep neural network architecture for speaker embedding models where similarity in embedded utterance vectors explicitly approximates the similarity in vocal patterns of speakers. The proposed architecture contains an additional speaker embedding lookup table to compute loss based on embedding similarities. Furthermore, we propose a new feature sampling method for data augmentation. Experimentation based on two databases demonstrates that our model is more effective at speaker identification and verification when compared to a fully connected classifier and an end-to-end verification model.

Index Terms: speaker classification, speaker discrimination, representation learning, loss combination, feature augmentation

1. Introduction

Representations that can distinguish speakers play essential roles in many speaker recognition-related problems like speaker identification, verification, detection, segmentation, and clustering as well as in speaker-dependent speech recognition systems. By comparing speaker representations, a system can recognize the identity of a speaker or verify whether or not the current speaker matches an enrolled target speaker.

To build speaker models, conventional approaches like i-vector [1] elaborate on decomposing feature space into subspaces corresponding to sound factors including speaker, session, and channel effects, and then extracting the speaker factors. With the recent success of deep models in representation learning, researchers have developed deep neural architectures (DNA) to generate speaker-specific representations [2, 3, 4]. One of the earlier studies introduced the d-vector approach [2], where a d-vector (or deep vector) has a similar role to i-vector but is derived from a deep neural network. Utterance-level d-vectors are obtained by averaging over the activations of the last hidden layer for all frame-level features.

One of the main criticisms on the d-vector approach in verification is that it handles the front-end representation learning and back-end scoring components separately. To address this issue, Heigold et al. [3] proposed an end-to-end architecture that jointly optimizes the system components in an integrated network for the "OK Google" benchmark of a text-dependent case where speakers are required to state a predetermined phrase. Later, Snyder et al. [5] extended the problem domain to text-independent speaker verification context by developing another end-to-end architecture. For Microsoft's "Hey, Cortana" benchmark, attention-based end-to-end models were introduced [6]. Recently, Baidu researchers presented an end-to-end neural speaker embedding system [4], which transforms utterance-level input into a speaker embedding represented as a continuous vector. Rather than training front-end and back-end components together, they trained the networks so that cosine similarity in the embedding space corresponds to utterance similarity as much as possible while assuming cosine similarity based back-end classifiers.

End-to-end systems were successful in their own benchmark tasks due to large amounts of dedicated data, but most of them suffer from difficulties in model training and lack of generalizability [7]. The difficulties come mainly from the fact that those systems take utterance pairs as training input to learn whether each pair is from the same speaker or different speakers. Propagating pairs through trainable subnetworks increases computational costs and the possibility of unstable convergence. Training the model in a pair-wised style highly depends on the distribution of the sampled pairs. In addition, the task-oriented nature of the end-to-end approach causes limited reusability while disregarding the embedding systems [4, 8].

This paper presents a DNA for speaker embedding models where the vector space similarity of utterance embeddings directly approximates the similarity of vocal patterns of speakers. We have similar motivation as [4] while proposing another loss term instead of using triplet loss. Our DNA does not take sampled pairs nor triplets as input in the training phase, avoiding the aforementioned difficulties. More specifically, the proposed method minimizes the additional loss that measures the errors based on the explicit computation of the similarities between the current embedding and updatable speaker embeddings in memory, which is inspired by word embedding models [9, 10, 11]. The computational burden of the additional term is marginal compared to that of the entire graph. Furthermore, we train the model on utterance-level speech segments instead of frame-level ones as in [3, 4, 5, 6, 8], since many speaker recognition applications, like user verification systems with certain pass phrases, perform tasks on the utterance-level.

We introduce an embedding DNA with its training procedure for speaker identification and verification in section 2. Section 3 provides the experimental results of the proposed method on TIMIT and LibriSpeech databases. Finally, the last section summarizes and concludes the paper.

2. Proposed Method

We first explain how to augment data for training speaker embedding models whose inputs are feature vector sequences ex-

10.21437/Interspeech.2018-1044

tracted from utterance-level speech segments. Next, we describe the baseline network structure for speaker classification and propose a generalized architecture. The speaker classifier itself can be employed for a closed-set speaker identification, in which a system is aware of all candidate speakers. Hence, we use two expressions, speaker classifier and speaker identification model, interchangeably, depending on context. After completing the classifier training, we reuse the trained network as a feature extractor for verification. In this case, we refer to the classifier training and the network adaptation as pre-training and fine-tuning, respectively.

2.1. Data Augmentation

Augmentation is crucial in utterance-level modeling where the number of training instances is far smaller than that in frame-level modeling, especially when there are a few utterances per speaker. While there have been some augmentation methods that were successful in training deep models for speech recognition [12, 13] and classifying a speaker's native language [14], the methods for speaker recognition were not thoroughly investigated yet. Since frame-level modeling [15, 16] itself could increase the number of training instances by extracting many frames from an utterance, there was less motivation to augment data. For utterance-level modeling, most successful systems trained their network with sufficient data from more than tens of thousands of speakers [3, 4, 5, 6]. Therefore, previous papers rarely addressed data augmentation problem.

Perturbing speech data without changing labels, is a common augmentation strategy in speech recognition. However, many perturbation methods that preserve phoneme labels are not speaker-invariant. Variations on frequency domain may involve a risk that perturbed features will lose the characteristics of the speaker [12, 17]. Tempo and speed perturbation [13] are also prone to such limitations because temporal durations for each phoneme engage deeply with different voice generation [18].

To preserve speaker labels, we follow a resampling style where each utterance is randomly split into several pieces. Then, a piece is either removed or left in an alternating fashion, and the remaining pieces are concatenated. This is an extension of the augmentation implemented in [14], which sampled continuous subsequences from the original input. We select the longer concatenated speech between the pieces at odd positions and the pieces at even positions in order to prevent the input from being too short. Since the procedure runs in each epoch and the deep learning process usually iterates on a training set over many epochs, the deleted speech in an iteration will be fed into a model in other iterations.

2.2. Speaker Classifier

The baseline network consists of a d-vector extraction module, which acts as a transform function $\delta(\cdot)$ that maps a sequence of feature vectors, x, to d-vector d, i.e. $\delta(x) = d$, and a fully connected layer whose output nodes correspond one-to-one with speaker IDs (Figure 1). We use a long short-term memory (LSTM) recurrent neural network (RNN) [19] with a single last output followed by a fully connected layer for extracting the d-vector as in [3]. While it might improve the model to replace the subnetwork structure with sophisticated ones, here we focus on formulating loss with the extracted d-vector rather than architecting the subnetwork for transforming a feature sequence into a d-vector. The trainable weights of the network are optimized using softmax cross entropy loss l_{f} with fully connected output

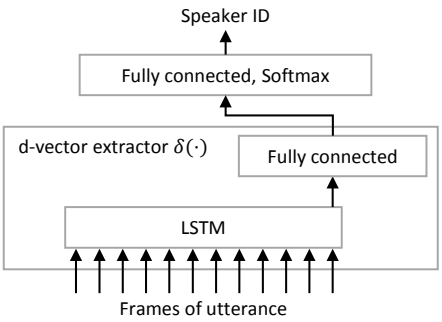

Figure 1: *Baseline Architecture.*

activations a_{f}, of which the i^{th} element is defined as follows:

$$(a_{\mathrm{f}})_i = W_i \cdot d + b_i, \tag{1}$$

where W_i and b_i are the weights and bias of the fully connected layer corresponding to speaker ID $i \in \{1, 2, ..., K\}$, respectively. The size of a_{f} equals to the number of speakers in the training set, K.

For tasks that involve handling out-of-domain speakers like speaker verification, δ can produce speaker representations for the input of a back-end classifier, after trained as a feature extractor by minimizing l_{f}. A straightforward approach to a back-end classifier is computing the cosine similarity between two representations for speaker discrimination problems that determine whether or not they are from the same speaker [2], but the reported performance was not satisfactory [3, 20]. Although other back-end classifiers including linear discriminant analysis (LDA) and probabilistic LDA were investigated [7, 21], there still remain inconsistencies in optimization objectives between classification and discrimination.

Another promising approach is to construct a Siamese network and to optimize the d-vector extraction module and threshold part simultaneously. The Siamese architecture consists of two identical subnetworks that share trainable weights [22]. The subnetworks transform speech segment pairs into two d-vectors, and then scalar energy function evaluates the compatibility among them. In [5], the authors used a Siamese network in training phase, and the models in [3] and [6] were also reduced to Siamese networks when the number of enrollment utterances is one.

However, these models require cumbersome procedures of sampling genuine and impostor pairs by picking the most similar impostor for each utterance [6] or constructing a mini-batch by selecting genuine pairs without any two pairs from the same speaker and then forming impostor pairs among them [5]. Recently, triplet loss was employed to build speaker embedding systems [4, 8], but the sampled triplets must satisfy the hard constraints. Moreover, training identical subnetworks at the same time resulted in unstable convergence [7]. For that reason, [4] pre-trained a softmax speaker classifier and [5] fed training pairs by a meticulous scheme that arranges the feeding order of utterances according to duration.

Figure 2 depicts our proposed architecture that contains a trainable embedding lookup table E of size $K \times h$ where h is the d-vector size, similarly to that of word embedding models [9, 10]. Each embedding in the table, $E_i \in \mathbb{R}^h$, acts as an enrolled speaker model of the corresponding speaker. We compute the similarity scores between the d-vector and speaker embeddings in the table. With this computation, the model can

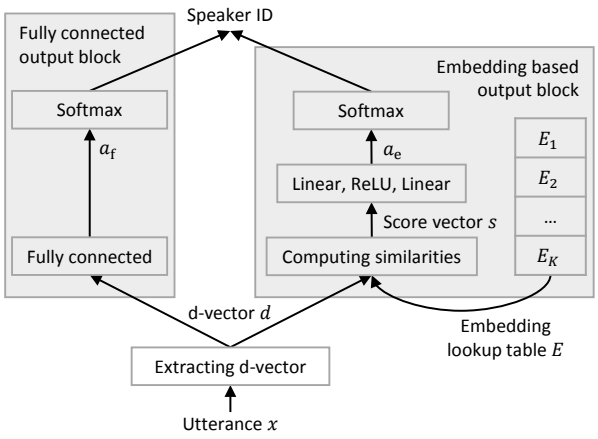

Figure 2: *Generalized Architecture. Data flow along arcs and go through operations expressed as box objects.*

compare d-vectors in an indirect way without requiring utterance pair input. The score vector is used to compute the softmax cross entropy loss. Just before the softmax layer, the score vector goes through a linear layer, a rectified linear unit (ReLU) layer, and then a linear layer once more. This explicit formulation stimulates the d-vectors of the same speaker as close as possible, whereas those of different speakers maintain distance from each other regarding the certain similarity metric. The linear and ReLU layers keep scores lower than a certain similarity to enlarge the influence of the speaker embeddings near the d-vector.

We used a cosine similarity metric $sim_{\cos}$ in this work, and the i^{th} element of the similarity score vector s is expressed as

$$s_i = sim_{\cos}(E_i, d) = \frac{E_i \cdot d}{\|E_i\|_2 \|d\|_2}. \tag{2}$$

s in (2) is identical to d in (1) except that the d-vector and embeddings are normalized to have unit Euclidean norm and that the bias term is omitted. This row normalization was reported to be consistently superior to the cases of non-normalization, column normalization only, and both row and column normalization in learning word embeddings [11]. The loss term based on the score vector is derived as

$$a_e = Linear(ReLU(Linear(s))), \tag{3}$$

$$l_e = -\log \sigma_y(a_e), \tag{4}$$

where $\sigma_k(\cdot)$ yields the k^{th} element value of softmax output and y is speaker ID. We generalize the baseline model by defining a combined loss as $(1-\lambda)l_f + \lambda l_e$, where $0 \leq \lambda \leq 1$. For classification, the predicted label is obtained as $\text{argmax}_k\,(1-\lambda)\sigma_k(a_f) + \lambda\sigma_k(a_e)$. Though we designed this architecture to bridge the gap between classification and discrimination, the classification performance itself was improved as well (see Section 3.2).

2.3. Fine-tuning for Verification

We followed the speaker verification protocol introduced in [3] by using only one enrollment utterance to simplify the discussion. Since the proposed classifier learns to map utterances to the embedding space where the cosine similarity reflects speaker characteristics well, it is sufficient to find a threshold level of the similarity score for verification. The score is computed as $sim_{\cos}(\delta(x_{\text{enrol}}), \delta(x_{\text{test}}))$ given the enrollment utterance x_{enrol} and test utterance x_{test}. Then simple logistic regression leads to a threshold mechanism just as in the end-to-end architecture [3]. We obtain the verification model by reusing the pre-trained weights of the d-vector extraction module from the classifier and fine-tuning only the logistic regression.

3. Experiments and Results

3.1. Experiment Setup

We prepared the TIMIT and LibriSpeech databases [23] containing 16kHz sampled recordings for our experiments, and divided speakers in each corpus into two groups: one for training and testing speaker classifiers as well as fine-tuning verification models and the other for only testing verification models. For TIMIT corpus, we assigned 462 speakers composing "train" data to the first group, while 168 speakers composing "test" data were reserved for the second group. For speaker classification, as in [16], we used "SX" and "SI" sentences in the first group for training and "SA" sentences in the first group for testing. Similarly, the LibriSpeech database was decomposed into two datasets: "train-clean-100" (251 speakers) and "train-clean-360" (921 speakers) for the first group as well as "dev-clean" (40 speakers) and "test-clean" (40 speakers) for the second group. Then, we split the former group of LibriSpeech randomly into training and test data with 9:1 ratio for speaker classification.

Each frame consists of 40 log Mel filterbank energy features extracted from a frame size of 32 ms with a frame shift of 16 ms. Note that the presented experiments were performed without any other speech preprocessing including voice activity detection, mean cepstral substitution, and global or speaker-level normalization with mean and variance. Instead, RNN is expected to conduct preprocessing to some extent, which follows the deep learning trend towards minimizing preprocessing of input.

The d-vector size and the number of LSTM units were 256 for the experimented models in this work. We limited the maximum length of feature vector sequence x to 200 frames, because too many steps for backpropagating-through-time are not desirable [19]. Truncating the sequence of features to the fixed length was done after the augmentation. In the training phase, we applied batch normalization and dropout of 0.1 for the fully connected layer followed by the last RNN output. The batch size was 256, and AdamOptimizer [24] updated the parameters with learning rate of 10^{-4}, $\beta_1 = 0.9$, and $\beta_2 = 0.99$ for all experiments. We regularized the baseline model with L2 loss through multiplying the average of the squared weights of the fully connected layers by the regularization parameter 0.01. Since the generalized architecture and the Siamese architecture normalize d-vectors during the cosine similarity computation, we did not apply L2 regularization for them.

3.2. Performance Comparison

To investigate the effect of the augmentation introduced in Section 2.1, we trained the baseline speaker classifiers on the TIMIT database by varying the number of splitting points p from zero (no augmentation) to five. The results in Figure 3 show the effectiveness of the proposed method. The dataset augmentation for the classification on LibriSpeech was not as effective as on TIMIT since the number of utterances per speaker in LibriSpeech is more than ten times as that in TIMIT.

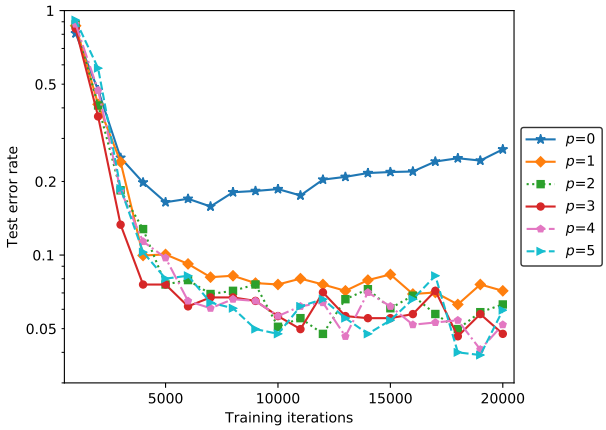

Figure 3: *Augmentation effect with respect to the number of splitting points experimented on TIMIT. $p = 0$ means no augmentation.*

Table 1: *Verification error rates (%) for different training schemes.*

Training scheme	**TIMIT**	**Libri**
without pre-training	9.2	10.4
fine-tuning the baseline ($\lambda = 0$)	4.4	5.1
fine-tuning the proposed ($\lambda = 0.5$)	3.6	4.5

Nevertheless, the augmentation did not degrade the performance and helped the models be robust. Therefore, we applied the augmentation using three splitting points for all the experiments presented below.

Figure 4 visualizes the speaker identification performance of three generalized models: the model with $\lambda = 0$ (called M0), which is the baseline model; the model with $\lambda = 1$ (called M1), which minimizes l_e without considering l_f; and the model with $\lambda = 0.5$ (called M0.5). The experiments were repeated several times and the error rate of M0 fluctuated between 3.1% and 5.7% while that of M0.5 was in the range of $1.7 \sim 3.1\%$ on TIMIT. On LibriSpeech, M0 and M0.5 achieved $2.3 \sim 4.3\%$ and $0.8 \sim 1.8\%$ error rates respectively. For both databases, M0.5 was consistently better than the others. The performance of M1 was poor at the early stage of the training, but it caught up with the baseline model performance at certain iterations. Minimizing the combined loss accelerated the training of M1 as well as enhanced the performance of M0.

Next, we investigated the d-vector extraction module of the proposed model learned from the speaker identification datasets to see if it was still useful in speaker verification tasks. For this investigation, we sampled genuine pairs and impostor pairs at 1:1 ratio from each group, as explained in the previous subsection. We did not sample impostor pairs across genders to avoid trivial instances. Table 1 presents the results for two databases by comparing the error rates of the Siamese model after training forty thousand iterations without pre-training with the two fine-tuned models, each of which reused the weights obtained after pre-training thirty thousand iterations of M0 and M0.5, respectively. Note that an iteration for pre-training takes less processing time than an iteration for training the Siamese network, because the Siamese network evaluates the activations of both inputs in a pair.

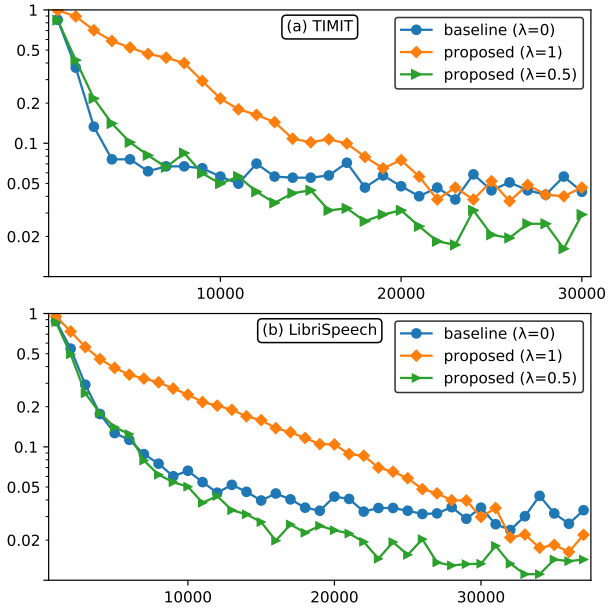

Figure 4: *Speaker identification results of the proposed models with respect to λ. The x-axis signifies the number of the training iterations, and the y-axis signifies the test classification error rate in log scale.*

The results show that the fine-tuned models outperformed the Siamese model without pre-training, which is in line with [7, 25]. Our results are similar to those of the small training set experiment in [3] rather than those of the large training set experiment in the same paper. These together imply that the performance of the Siamese model is more sensitive to the number of speakers in training dataset than the fine-tuned models. In addition, the fine-tuned model that reused M0.5 was better among the fine-tuned models for both databases, which confirms the effectiveness of the proposed method.

4. Conclusion

In this paper, we presented a tractable and effective method for training speaker embedding networks by minimizing an additional loss based on embedding similarities. The experiments showed that our model outperformed the fully connected baseline network model for speaker identification. For speaker verification, the proposed method achieved better results than the model that fine-tuned the baseline classifier or the original d-vector approach [2], and outperformed the model based on the Siamese network, which is a simplified version of the end-to-end approach [3]. Furthermore, the data augmentation was essential when the speaker labeled instances of utterance-level input were not sufficient enough to learn deep representation.

5. Acknowledgements

This work was supported by Kakao and Kakao Brain corporations.

6. References

[1] 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.

[2] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 4052–4056.

[3] G. Heigold, I. Moreno, S. Bengio, and N. Shazeer, "End-to-end text-dependent speaker verification," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5115–5119.

[4] C. Li, X. Ma, B. Jiang, X. Li, X. Zhang, X. Liu, Y. Cao, A. Kannan, and Z. Zhu, "Deep speaker: an end-to-end neural speaker embedding system," *arXiv preprint arXiv:1705.02304*, 2017.

[5] D. Snyder, P. Ghahremani, D. Povey, D. Garcia-Romero, Y. Carmiel, and S. Khudanpur, "Deep neural network-based speaker embeddings for end-to-end speaker verification," in *2016 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2016, pp. 165–170.

[6] S.-X. Zhang, Z. Chen, Y. Zhao, J. Li, and Y. Gong, "End-to-end attention based text-dependent speaker verification," in *2016 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2016, pp. 171–178.

[7] D. Wang, L. Li, Z. Tang, and T. F. Zheng, "Deep speaker verification: Do we need end to end?" in *Proceedings of APSIPA Annual Summit and Conference*, vol. 2017, 2017, pp. 12–15.

[8] H. Bredin, "Tristounet: Triplet loss for speaker turn embedding," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 5430–5434.

[9] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[10] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *EMNLP*, vol. 14, 2014, pp. 1532–1543.

[11] O. Levy, Y. Goldberg, and I. Dagan, "Improving distributional similarity with lessons learned from word embeddings," *Transactions of the Association for Computational Linguistics*, vol. 3, pp. 211–225, 2015.

[12] N. Jaitly and G. E. Hinton, "Vocal tract length perturbation (vtlp) improves speech recognition," in *Proc. ICML Workshop on Deep Learning for Audio, Speech and Language*, 2013, pp. 625–660.

[13] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *INTERSPEECH*, 2015, pp. 3586–3589.

[14] G. Keren, J. Deng, J. Pohjalainen, and B. W. Schuller, "Convolutional neural networks with data augmentation for classifying speakers' native language," in *INTERSPEECH*, 2016, pp. 2393–2397.

[15] K. Chen and A. Salman, "Learning speaker-specific characteristics with a deep neural architecture," *IEEE Transactions on Neural Networks*, vol. 22, no. 11, pp. 1744–1756, 2011.

[16] Z. Ge, A. N. Iyer, S. Cheluvaraja, R. Sundaram, and A. Ganapathiraju, "Neural network based speaker classification and verification systems with enhanced features," *arXiv preprint arXiv:1702.02289*, 2017.

[17] L. Lee and R. Rose, "A frequency warping approach to speaker normalization," *IEEE Transactions on speech and audio processing*, vol. 6, no. 1, pp. 49–60, 1998.

[18] S. Arik, G. Diamos, A. Gibiansky, J. Miller, K. Peng, W. Ping, J. Raiman, and Y. Zhou, "Deep voice 2: Multi-speaker neural text-to-speech," *arXiv preprint arXiv:1705.08947*, 2017.

[19] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *INTERSPEECH*, 2014.

[20] H.-S. Lee, Y.-D. Lu, C.-C. Hsu, Y. Tsao, H.-M. Wang, and S.-K. Jeng, "Discriminative autoencoders for speaker verification," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 5375–5379.

[21] N. Chen, Y. Qian, and K. Yu, "Multi-task learning for text-dependent speaker verification," in *INTERSPEECH*, 2015.

[22] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a" siamese" time delay neural network," in *Advances in Neural Information Processing Systems*, 1994, pp. 737–744.

[23] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.

[24] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[25] G. Bhattacharya, J. Alam, T. Stafylakis, and P. Kenny, "Deep neural network based text-dependent speaker recognition: Preliminary results," in *Odyssey: the Speaker and Language Recognition Workshop*, 2016, pp. 9–15.