OPTIMIZING NEURAL NETWORK EMBEDDINGS USING PAIR-WISE LOSS FOR TEXT-INDEPENDENT SPEAKER VERIFICATION

Hira Dhamyal, Tianyan Zhou, Bhiksha Raj, Rita Singh

Carnegie Mellon University

ABSTRACT

Recently neural networks have pushed state of the art performances on the speaker verification task, on top of the i-vector baseline systems. The key in these systems is to extract speaker discriminative features from the utterances. In this paper, we study the use of a new loss function, called Pair-Wise loss. We hypothesize that this loss optimizes the neural network to perform better on instances outside of the training set; i.e makes the network more generalizable. We evaluate this loss on the speaker verification problem. Pair-Wise loss explicitly tries to increase the gap between the similarity score distributions from the same class and different class pairs. In other words, instead of minimizing the classification error, the loss minimizes the area under the Detection Error Trade-off (DET) curve. Experimental results emphasize our stated hypothesis. We also achieve an overall improvement of 30% EER over the i-vector baseline in the SRE10 dataset. In addition, the features extracted from this neural network are complementary to i-vectors and their combination can further improve the i-vector baseline.

Index Terms: speaker verification, pair-wise loss, residual networks, i-vectors

1. INTRODUCTION

“Speaker verification” refers to the task of verifying a speaker’s claimed identity from their voice. This is often recast as the problem of comparing pairs of speech recordings: given two speech recordings (one of which is known to belong to the claimed identity and the other is the “verification” recording from the subject claiming this ID), determine if they are spoken by the same person. Typically, the task is *text independent*, *i.e.* there is no expectation that the words spoken in two recordings are the same, *e.g.* [1].

Traditionally, this task has been performed using *i-vectors*. I-vectors are factor-analyzed fixed-length vector characterizations of the distribution of spectral features derived from a speech signal [2], which are usually further transformed through probabilistic linear discriminant analysis (PLDA) to make them more speaker discriminative [3, 4, 5, 6]. The similarity of two recordings is estimated through a statistical hypothesis test. If the similarity score exceeds a threshold, the two recordings are labelled as a “match,” *i.e.* to be from the same speaker.

An alternate approach is to use neural networks to extract speaker-discriminative features from speech recordings. One way of doing so is to train neural networks to classify a large number of speakers. Subsequently, the final classification layer of the network is removed, and the remaining network is employed as a feature extractor. With sufficient training, the feature vectors extracted from the network are expected to generalize to discriminate speakers outside the training set. Recent work using numerous variants of this approach show competitive results for speaker verification [7, 8, 9, 10, 11, 12].

Naïvely trained neural network feature extractors do not, however, result in the best features for speaker verification. One reason is that neural-network approaches emphasize the separation of the *distributions* of feature vectors for different speakers, but do not directly optimize for the *direct comparison* of feature vectors. This can be suboptimal from the point of view of speaker verification, as illustrated in Figure 1. While the distributions of feature vectors for different speakers may indeed be well separated, the distance between two recordings for the same speaker may be greater than the distance between recordings of different speakers, resulting in erroneous inferences about speaker match. Attempts to address this

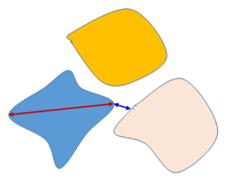

Fig. 1: The clusters represent speakers. While the clusters are well separable by a linear classifier, the distance between two farther instances within the blue cluster (red arrow) is much greater than the distance between two close instances from different clusters (blue arrow).

issue generally try to explicitly increase the separation between the scatters of features computed for different speakers, through modified loss functions such as center loss [13] and angular loss [14]. These, however, depend on the expectation that the increased separation will not only be sufficient to overcome the problem illustrated in Figure 1 for speakers in the training set, but will also generalize to novel, previously unseen speakers in the test data, an expectation that does not always hold up.

An alternate, possibly more appropriate approach is to directly optimize the network for the *comparison* of speech recordings. This is generally performed through the triplet loss [15], minimization of which attempts to make (features derived from) any recording for a speaker more similar to other recordings by the same speaker than to any imposter. While this does improve the discriminativeness of features derived from the network, we contend that it still misses a key issue. The real objective desired in any kind of “similarity based matching” problem is to minimize the overlap between the *distribution* of similarity scores under match and mismatch (as we explain in Section 2). Directly doing so may be expected to result in better generalization than merely maximizing the distinction between matches and mismatches for individual speakers.

In this paper, we propose a new loss function called the “Pair-Wise” loss, which embodies this principle. Pair-wise loss attempts to directly increase the gap between the distributions of similarity scores of matched and mismatched pairs of input. To formulate the optimization we explicitly consider the types of errors in speaker verification, *i.e.* missed detections and false alarms. To make a decision for a given pair of speech recordings, a similarity score computed between their features is compared to a threshold. The choice of the threshold represents a trade-off between missed detection and false alarm rates. Ideally, there would be a threshold where both false alarms and missed detections are zero. This happens when the distribution of similarity scores for matched recordings has no overlap with that of similarity scores for mismatched recordings. More generally, reduced verification errors may be obtained by minimizing the overlap between the distributions. This is the basis of our work.

The pair-wise loss significantly improves results over other neural-net based feature extraction methods for speaker verification. Additionally, when fused with i-vectors, it results in a nearly 30% improvement over the i-vector baseline.

The rest of the paper is organized as follows: Section 2 outlines the basic problem statement. Section 3 describes our proposed approach and the pair-wise loss function. Section 4 describes our network architecture and settings. Finally, in sections 5 and 6, we present our experimental results and conclusions.

2. BACKGROUND AND RELATED WORK

The problem of speaker verification can be formulated as follows: given two recordings $\mathbf{x}_1$ and $\mathbf{x}_2$, we must determine if both of them were spoken by the same speaker or not. Formally, representing the event that they are from the same speaker by $\mathcal{H}_s$ and from different speakers by $\mathcal{H}_d$, we must determine whether $(\mathbf{x}_1, \mathbf{x}_2) \in \mathcal{H}_s$ or if $(\mathbf{x}_1, \mathbf{x}_2) \in \mathcal{H}_d$.

To do so, we derive a fixed-length *feature* $f_\mathbf{x}$ from each recording and compute a similarity score $S(f_{\mathbf{x}_1}, f_{\mathbf{x}_2})$ between them, typically either the log likelihood ratio under the hypotheses $\mathcal{H}_s$ and $\mathcal{H}_d$ [5], or the cosine similarity between $f_{\mathbf{x}_1}$ and $f_{\mathbf{x}_2}$ [16]. If this similarity exceeds a threshold θ, we decide that the two recordings "match," *i.e.* they belong to the same speaker, otherwise we declare a "mismatch."

The accuracy of this procedure is critically dependent on the features $f_\mathbf{x}$ derived from the recordings, which must be speaker discriminative. The key metric is the similarity computed between recordings. Figure 2 shows a probability distributions of similarity scores under match and mismatch, $P(S(f_{\mathbf{x}_1}, f_{\mathbf{x}_2})|\mathcal{H}_s)$ and $P(S(f_{\mathbf{x}_1}, f_{\mathbf{x}_2})|\mathcal{H}_d)$ respectively. The overlap between the two represents the "region of confusion" – recording pairs whose similarity score falls in this region are likely to be misclassified. The fraction of all tests that result in an erroneous outcome depends directly on the overlap area. Hence, to maximize verification accuracy, the *features* $f_\mathbf{x}$ derived from the recordings must be such that the two distributions are well separated.

Ideally, the function that extracts features from speech signals must aim to separate the distributions of *similarity scores* under match $\mathcal{H}_s$ and mismatch $\mathcal{H}_d$. Our objective is to develop a neural-network based feature extractor that explicitly attempts to maximize this separation. We do so through the pair-wise loss.

The closest related approaches (*e.g.* [17]) train a neural network feature extractor with the *triplet* loss, which maximizes the gap between the similarity of individual "anchor" recordings for a speaker to matched recordings from themselves and the similarity of the anchor to the nearest competing speaker. In contrast, in maximizing the gap between the *distributions* of similarity scores under $\mathcal{H}_s$ and $\mathcal{H}_d$, the pair-wise loss function maximizes the gap between the similarity of matched pairs of recordings for a speaker and that of *any* mismatched pair of recordings for *any* two speakers. In effect, it is even more conservative than the triplet loss. Practically, while the triplet loss considers *triples* of recordings in each instance of training, the pair-wise loss considers *quadruples*.

The exact framework we use is elucidated in the next section.

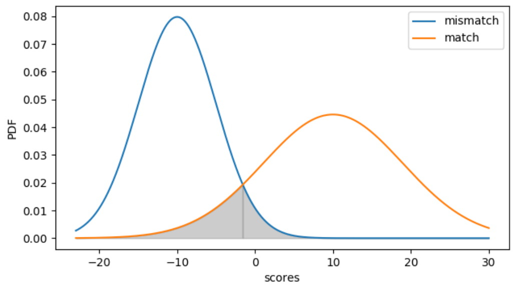

Fig. 2: Probability distributions of similarity scores under match and mismatch.

3. FEATURE LEARNING THROUGH PAIR-WISE LOSS

Let $F(\mathbf{x}; \Phi)$ with parameter Φ be the function that computes the feature $f_\mathbf{x}$ from a recording $\mathbf{x}$, *i.e.* $f_\mathbf{x} = F(\mathbf{x}; \Phi)$. Our objective is to learn $F(\mathbf{x}; \Phi)$ such that it minimizes the overlap between $P(S(f_{\mathbf{x}_1}, f_{\mathbf{x}_2})|\mathcal{H}_s)$ and $P(S(f_{\mathbf{x}_1}, f_{\mathbf{x}_2})|\mathcal{H}_d)$. Let $S_s \sim P(S|\mathcal{H}_s)$ and $S_d \sim P(S|\mathcal{H}_d)$ represent draws from the probability distributions of similarity scores under match and mismatch. The overlap area is equal to $P(S_s < S_d)$. This area must be minimized.

Formally, let $(\mathbf{x}_1, \mathbf{x}_2) \in \mathcal{H}_s$ and $(\mathbf{y}_1, \mathbf{y}_2) \in \mathcal{H}_d$ be random pairs of matched and mismatched recordings respectively. Let $S_X = S(f_{\mathbf{x}_1}, f_{\mathbf{x}_2})$ be the similarity computed between the matched pair $\mathbf{x}_1$ and $\mathbf{x}_2$, and similarly let $S_Y = S(f_{\mathbf{y}_1}, f_{\mathbf{y}_2})$ be the similarity computed between the mismatched pair $\mathbf{y}_1$ and $\mathbf{y}_2$. In order to optimize the feature extraction function $F(\mathbf{x}; \Phi)$ we must estimate Φ as

$$\hat{\Phi} = \arg\min_{\Phi} P(S_X < S_Y) \tag{1}$$

$P(S_X < S_Y)$ cannot be known, but an unbiased empirical estimator for it can be computed using the Wilcoxon Mann Whitney (WMW) statistic [18] on collections of randomly drawn pairs of matched and mismatched inputs. Representing the i^{th} randomly drawn matched pair as $X_i = (\mathbf{x}_1^i, \mathbf{x}_2^i)$ and the similarity score computed from it as S_{X_i}, and the j^{th} randomly drawn *mis*matched pair as $Y_j = (\mathbf{y}_1^j, \mathbf{y}_2^j)$ and the corresponding similarity score as S_{Y_j}, the estimate is

$$P(S_s < S_d) \approx \frac{1}{NM} \sum_i^N \sum_j^M \mathcal{I}(S_{X_i} < S_{Y_j})$$

where N and M correspond to the number of matched and mismatched pairs respectively and $\mathcal{I}()$ is the indicator function.

In principle, the estimator

$$\hat{\Phi} = \arg\min_{\Phi} \frac{1}{NM} \sum_i^N \sum_j^M \mathcal{I}(S_{X_i} < S_{Y_j}) \tag{2}$$

will give us a feature extractor $F(\mathbf{x}; \Phi)$ that minimizes the distribution overlap. In practice, the WMW statistic as given above only minimizes the *expected* value of $\mathcal{I}(S_{X_i} < S_{Y_j})$. To maximize the generalization of the derived features, we must conservatively optimize the *worst* case, rather than the average case, i.e. we must maximize the expected gap between the similarity score for a match and a *worst case* similarity score for mismatches.

We therefore define the following extreme value statistic

$$S^K_{Y,max} = max(S_i \sim P(S|\mathcal{H}_d),\ i = 1..K)$$

where S^K_{max} is the maximum of K random draws from $P(S|\mathcal{H}_d)$. We can now define an empirical loss function:

$$L(\Phi) = \frac{1}{N}\sum_i^N \mathcal{I}(S_{X_i} < S^K_{Y_i,max}) \quad (3)$$

where $S^K_{Y_i,max}$ is the largest of the similarity scores obtained from a random draw of K mismatched pairs, that are specific to the i^{th} matched pair. In practice, the indicator function is not differentiable, so we approximate it as a sigmoid. This gives us our final loss function:

$$L(\Phi) = \frac{1}{N}\sum_i^N \sigma(S^K_{Y_i,max} - S_{X_i}) \quad (4)$$

$$\hat{\Phi} = \arg\min_{\Phi} L(\Phi) \quad (5)$$

Note that the sigmoid in Equation 4 can potentially be replaced by other monotonic functions such as a RELU, ELU or leaky RELU in this setting. These may be viewed as increasing the margin between the distributions.

4. SYSTEM IMPLEMENTATION

We implement the feature extractor $F(\mathbf{x}; \Phi)$ as a convolutional neural network that operates on spectrograms derived from the audio signal. The similarity score we optimize is the cosine similarity. We present the details below.

4.1. Parameterizing the speech signal

The input speech recordings are initially parametrized into a mel-frequency spectrogram, comprising sequences of log mel spectral vectors (melspec) [19]. The parametrization uses a bank of 63 mel-frequency filters. A frame length of 25ms and a frame shift of 10ms are used for the parametrization in our experiments. Non-speech regions in recordings are excised using an energy-based voice-activity detector [20]. The remaining vectors are mean normalized. During training each recording length is restricted to a randomly selected contiguous block of 16,383 frames, to conform to the logistical needs of batch processing. At testing time, entire recordings are used for the feature extraction.

4.2. Neural-network feature extractor

Our feature extractor is a convolutional neural network (CNN), with a residual network structure [21]. The network comprises a set of five "naïve" convolutional layers, augmented by several residual blocks. A residual block contains two convolutional layers in our experiments – this was found to be optimal.

Table 1 shows the configuration detail of our system. On top of a 5-layer CNN, we add two residual blocks after each of the first three convolutional layers. All convolutional layers are followed by a batch normalization layer and use rectified linear unit (ReLU) activation. The width of the output of the final convolutional layer is proportional to the length of the input speech recording, the duration of which may vary. In order to obtain a fixed length feature vector, we apply temporal average pooling to the entire feature map in the final layer. The final feature vector obtained has 128 dimensions.

Table 1: Deep neural network configurations (notation for convolutional layer: (channel, kernel, stride, padding)).

Setting	Detail
ResNet	Conv: (4, 3×3, 2, 0)
	Residual block: (4, 3×3, 1, 1)
	Residual block: (4, 3×3, 1, 1)
	Conv: (16, 3×3, 2, 0)
	Residual block: (16, 3×3, 1, 1)
	Residual block: (16, 3×3, 1, 1)
	Conv: (64, 3×3, 2, 0)
	Residual block: (64, 3×3, 1, 1)
	Residual block: (64, 3×3, 1, 1)
	Conv: (256, 3×3, 2, 0)
	Conv: (128, 3×3, 2, 0)
	temporal average pooling: entire feature map
Initialization	FC: classification layer
	Softmax
	Cross entropy loss
Fine tuning	Pair-wise loss

4.3. Training the network

Initializing the network well is critical. We initialize the network by adding a softmax classification layer at the top to classify between all the speakers in the training set. This is trained for several epochs (70 in our experiments) to minimize the cross-entropy loss. Thereafter we remove the final classification layer and switch to the proposed pair-wise loss, and continue training for several additional epochs. We use stochastic gradient descent (SGD) with mini-batches of size 128. Momentum and weight decay are set to 0.9 and 0.0001 respectively which was found to be optimal.

4.4. Sampling for mini-batches

We organize the sampling of mini-batches as follows. Within each minibatch, we first choose P *matched* pairs of recordings, from P randomly chosen speakers. Subsequently, for each of the P pairs, we randomly draw K *mismatched* pairs of recordings. In our experiments we set $K = 40$. From these K recordings, we select the pair with the highest similarity score. The matched and mismatched scores together then feed into our loss. Note that the result is that for each entry into the loss function, *four* speech signals are considered. In our experiments, P was set to 32, resulting in 128 speech recordings per minibatch. Speakers and matched pairs are not sampled with replacement. All other sampling is with replacement.

5. EXPERIMENTS

5.1. Data description

For training, we use the dataset published by the NIST speaker recognition evaluations (SRE). Our training set consists of several previous SRE sets from year 2004, 2005, 2006 and 2008, with a total of 36612 utterances and 3805 speakers.

Our evaluation set is SRE10 consisting of 7169 match pairs and 408950 many mismatch pairs. It provides two sets of recordings, one designated as enrollment and the other as testing. Test pairs are chosen by picking one utterance from the enrollment set and one from the testing set separately. If the two utterances are spoken by the same speaker, it's a matched pair, else it's a mismatched pair. A trial file specifies 416,119 pairs with match/mismatch labels. Our experimental results are reported on these trials.

For our i-vector comparator, we implemented the i-vector baseline system based on the Kaldi SRE10 recipe [22].

5.2. Results

As noted in Section 4, the loss function in Equation 4 can employ a variety of approximations for the indicator function, not just the sigmoid. We test different approximation functions in our loss including the Sigmoid, ELU and the LeakyReLU. All networks are identically initialized. While the actual loss is optimized using the cosine similarity metric, final verification employs PLDA scoring, similarly to the i-vector baseline. The EERs for the Sigmoid, ELU and LeakyRelu activations are 2.846, 2.86 and 3.125 respectively.

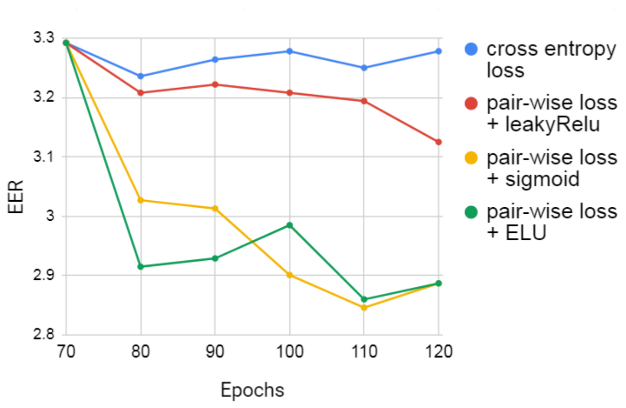

Fig. 3: EER for cross-entropy loss and pair-wise loss.

We also compare multiple approaches. Firstly, we continue training the initial CNN with the cross-entropy loss (with the final classification softmax layer) for an additional 50 epochs. This corresponds to the traditional approach of using CNNs for feature extraction [17]. In the comparators, we switch to optimizing using the proposed pair-wise loss (after removing the softmax layer) after the preliminary 70 initial iterations with the cross-entropy loss. Figure 3 demonstrates the EER trend for cross-entropy loss and pair-wise loss. We observe that cross entropy doesn't improve performance and even seems to result in overfitting. By contrast, our loss decreases EER, especially for the ELU and Sigmoid approximations, implying that the pair-wise loss can learn speaker representative features and perform better in speaker verification tasks.

Secondly, we compare pair-wise loss with triplet loss. The network architecture and initialization are identical to those used for the pair-wise loss. The CNN is also optimized on the triplet loss using the best performing margin value. An EER of 3.32% is achieved with the margin of 0.2. As Figure 4 shows, the pair-wise loss, with its EER of 2.88% outperforms triplet loss significantly.

Thirdly, we compare our approach with CLNet as explained in the next sections proposed in [23], which were shown to achieve the best results for neural-network based feature extraction when used as an ensemble of three nets. As shown in Figure 4, our method greatly outperforms a single CLNet. For the sake of fairness, we have not compared our model to the CLNet ensemble, since our own network is currently not an ensemble.

5.3. Feature and Score fusion

Generally speaking, the fusion of several complementary systems is effective to further improve the performance. Therefore, we also try to combine our features with i-vectors. For a given input utterance, we have 128-dimensional feature from our neural network and a 600-dimensional i-vector from the baseline system. After length normalization, we concatenate the two features and perform PLDA scoring (early fusion). We also perform combination by averaging the scores obtained independently by the i-vector system and our system (late fusion). Figure 4 presents the DET curves of the i-vector baseline system, our best system, triplet loss neural network and the two fusion systems. The best result is from early fusion, which achieves an EER of 1.911 and outperforms the i-vector baseline by 29.5%.

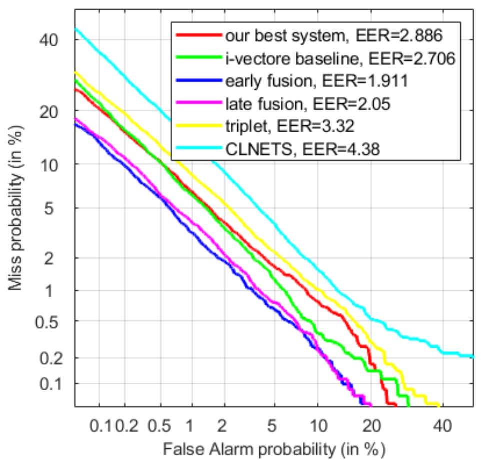

Fig. 4: DET curves of i-vector baseline system, triplet loss(m=0.2), CLNets, our best system and two fusion systems.

6. CONCLUSIONS

In this paper, we propose a pair-wise loss function which explicitly maximizes the distance between the similarity-score distributions for matched and mismatched pairs of recordings to improve speaker verification performance. Experimental results indicate that our loss function achieves better results in terms of separating score distributions compared to traditional methods of training neural networks to extract features for speaker verification. When fused with i-vectors, it is also observed to improve over the state-of-art i-vector baseline. We note several avenues for future work, including the investigation of different sampling strategies and similarity metrics to optimize the network. Finally, as shown in [24], harnessing a full-blown speech recognition system for feature extraction can greatly improve performance over simpler models that do not consider the content of the speech. While the logistical overhead may be undesirably high, we believe it to nonetheless be an avenue worth pursuing further.

7. REFERENCES

[1] "Nist 2016 speaker recognition evaluation plan," 2016.

[2] N. Dehak, P. Kenny, R. Dehak, P. Ouellet, and P. Dumouchel, "Front-end factor analysis for speaker verification," in *IEEE Transactions on Audio, Speech, and Language Processing*, 2011, pp. 788–798.

[3] S. Prince and J. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *IEEE 11th International Conference on Computer Vision*, 2007, pp. 1–8.

[4] J. A. Villalba and N. Brummer, "Towards fully bayesian speaker recognition: Integrating out the between-speaker covariance," in *INTERSPEECH 2011, Conference of the International Speech Communication Association, Florence, Italy, August*, 2011, pp. 505–508.

[5] D. Garcia-Romero and C. Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems," in *INTERSPEECH 2011, Conference of the International Speech Communication Association, Florence, Italy, August*, 2011, pp. 249–252.

[6] D. Garcia-Romero, X. Zhou, and C. Espy-Wilson, "Multicondition training of gaussian plda models in i-vector space for noise and reverberation robust speaker recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2012, pp. 4257–4260.

[7] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, "Deep neural network embeddings for text-independent speaker verification," in *INTERSPEECH*, 2017, pp. 999–1003.

[8] L. Li, Y. Chen, Y. Shi, Z. Tang, and D. Wang, "Deep speaker feature learning for text-independent speaker verification," in *INTERSPEECH*, 2017, pp. 1542–1546.

[9] S. Ranjan and J. H. L. Hansen, "Improved gender independent speaker recognition using convolutional neural network based bottleneck features," in *INTERSPEECH*, 2017, pp. 1009–1013.

[10] J. Jorrin, P. Garcia, and L. Buera, "Dnn bottleneck features for speaker clustering," in *INTERSPEECH*, 2017, pp. 1024–1028.

[11] D. Snyder, D. Garcia-Romero, and D. Povey, "Time delay deep neural network-based universal background models for speaker recognition," in *Automatic Speech Recognition and Understanding*, 2016, pp. 92–97.

[12] 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, Nov 2011.

[13] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *ECCV*, 2016, p. 499515.

[14] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "Sphereface: Deep hypersphere embedding for face recognition," *CoRR*, vol. abs/1704.08063, 2017. [Online]. Available: http://arxiv.org/abs/1704.08063

[15] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," *Journal of Machine Learning Research*, vol. 10, pp. 207–244, 2009.

[16] N. Dehak, R. Dehak, P. Kenny, N. Brummer, P. Ouellet, and P. Dumouchel, "Support vector machines versus fast scoring in the low-dimensional total variability space for speaker verification," in *Tenth Annual conference of the international speech communication association*, 2009, pp. 1559–1562.

[17] 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.

[18] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *Ann. Math. Statist.*, vol. 18, no. 1, pp. 50–60, 03 1947. [Online]. Available: https://doi.org/10.1214/aoms/1177730491

[19] S. B. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," in *Readings in speech recognition*. Elsevier, 1990, pp. 65–74.

[20] J. Ramirez, J. M. Górriz, and J. C. Segura, "Voice activity detection. fundamentals and speech recognition system robustness," in *Robust speech recognition and understanding*. InTech, 2007.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," pp. 770–778, 2015.

[22] 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 *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, Dec. 2011, iEEE Catalog No.: CFP11SRW-USB.

[23] Y. Wen, T. Zhou, R. Singh, and B. Raj, "A corrective learning approach for text-independent speaker verification," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2018.

[24] S. O. Sadjadi, S. Ganapathy, and J. W. Pelecanos, "The ibm 2016 speaker recognition system," 2016.