Interspeech 2018
2-6 September 2018, Hyderabad

Analysis of Length Normalization in End-to-End Speaker Verification System

Weicheng Cai[2], *Jinkun Chen*[2], *Ming Li*[1]

[1]Data Science Research Center, Duke Kunshan University, Kunshan, China
[2]School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou, China

ming.li369@dukekunshan.edu.cn

Abstract

The classical i-vectors and the latest end-to-end deep speaker embeddings are the two representative categories of utterance-level representations in automatic speaker verification systems. Traditionally, once i-vectors or deep speaker embeddings are extracted, we rely on an extra length normalization step to normalize the representations into unit-length hyperspace before back-end modeling. In this paper, we explore how the neural network learns length-normalized deep speaker embeddings in an end-to-end manner. To this end, we add a length normalization layer followed by a scale layer before the output layer of the common classification network. We conducted experiments on the verification task of the Voxceleb1 dataset. The results show that integrating this simple step in the end-to-end training pipeline significantly boosts the performance of speaker verification. In the testing stage of our L_2-normalized end-to-end system, a simple inner-product can achieve the state-of-the-art.
Index Terms: speaker verification, length normalization, end-to-end, deep speaker embedding

1. Introduction

Speaker recognition (SR) task can be defined as an utterance-level "sequence-to-one" learning problem. It is problem in that we are trying to retrieve information about an entire utterance rather than specific word content [1]. Moreover, there is no constraint on the lexicon words thus training utterances and testing segments may have completely different contents [2]. Therefore, given the input speech data, the goal may boil down to transform them into utterance-level representations, among them the inter-class variability is maximized and simultaneously the intra-class variability is minimized [3].

Typically, SR can be categorized as speaker identification (SID) task and speaker verification (SV) task [4]. The former classifies a speaker to a specific identity, while the latter determines whether a pair of utterances belongs to the same person. For the open-set protocol, speaker identities in the testing set are usually disjoint from the ones in training set, which makes the SV more challenging yet closer to practice. Since it is impossible to classify testing utterances to known identities in training set, we need to map speakers to a discriminative feature space. In this scenario, open-set SV is essentially a metric learning problem, where the key is to learn discriminative large-margin speaker embeddings.

There are generally two categories commonly used to obtain utterance-level speaker representations. The first consists of series of separated statistical models. The represent is the classical i-vector approach [5]. Firstly, frame-level feature sequences are extracted from raw audio signals. Then, selected feature frames in training dataset are grouped together to estimate a Gaussian Mixture Model (GMM) based universal background model (UBM) [6]. Sufficient statistics of each utterance on the UBM is accumulated, and a factor analysis based i-vector extractor is trained to project the statistics into a low-rank total variability subspace [5].

The other category relies on a model trained by a downstream procedure through end-to-end deep neural network [7, 8, 9, 10]. First, in the same way as the i-vector approach, frame-level feature sequences are extracted as well. Then an automatic frame-level feature extractor such as convolution neural network (CNN) [8, 11], time-delay neural network (TDNN) [9] or Long Short Term Memory (LSTM) network [7, 12] is designated to get high-level abstract representation. Afterward, a statistic pooling [9] or encoding layer [13] is built on top to extract the fixed-dimensional utterance-level representation. This utterance-level representation can be further processed by fully-connected (FC) layer, and finally connected with an output layer. All the components in the end-to-end pipeline are jointly learned with a unified loss function.

In classical i-vector approach, an extra length normalization step is necessary to normalize the representations into unit-length hyperspace before back-end modeling [14]. When it turns into end-to-end system, once we have extracted deep speaker embeddings from theneural network, such as x-vector [15], this length normalization step is also required when calculating pairwise scores.

In this paper, we explore end-to-end SV system where length normalization step is built-in inherently within the deep neural network. Therefore, the neural network can learn speaker embeddings being length-normalized in an end-to-end manner.

2. Related works

2.1. Length normalization in i-vector approach

Length normalization has been analyzed and proved to be an effective strategy for SR, but limited in conventional i-vector approach [14]. As demonstrated in Fig. 1, this simple non-linear transformation on i-vector has been the de facto standard before

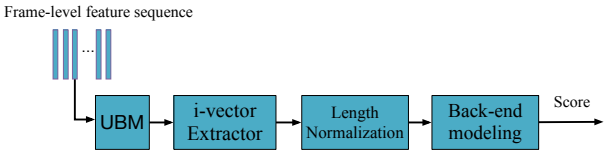


Figure 1: *Length normalization in conventional i-vector system*

This research was funded in part by the National Natural Science Foundation of China (61401524,61773413), Natural Science Foundation of Guangzhou City (201707010363), Science and Technology Development Foundation of Guangdong Province (2017B090901045), National Key Research and Development Program (2016YFC0103905).

10.21437/Interspeech.2018-92

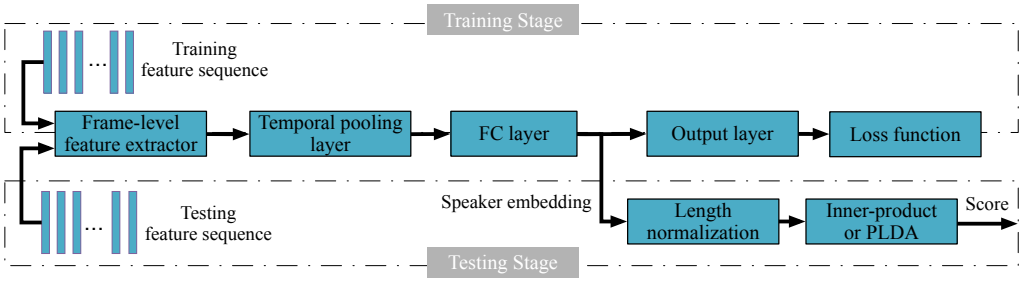


Figure 2: *Length normalization in common deep speaker embedding system.*

back-end modeling [16, 17].

For closed-set SID task, length normalization followed by logistic regression or support vector machine is commonly adopted to get the posterior probabilities for the speaker categories. For open-set SV task, cosine similarity or length normalization followed by probabilistic linear discriminant analysis (PLDA) scoring [18, 19] modeling is widely used to get the final pairwise scores. The cosine similarity is a similarity measure which is independent of magnitude, it can be seen as the length-normalized version of inner-product of two vectors. In these above systems, front-end i-vector modeling, length normalization step, and back-end modeling are all independent of each other and performed separately.

2.2. Length normalization in end-to-end system with triplet loss

Some previous works in [8, 12, 20] introduced triplet loss [21] and successfully trained models with the features being normalized in an end-to-end fashion. They all explicitly treat the open-set SV task as a metric learning problem. This kind of triplet loss approach naturally requires length normalization step to compute the distance of normalized unit vectors.

However, a neural network trained with triplet loss requires carefully designed triplet mining procedure. This procedure is non-trivial, both time-consuming and performance-sensitive [22]. Besides, many closed-set tasks like SID are equal to classification problem, it is intuitively not necessary to implement triplets mining procedure and explicitly treat them as metric learning problem. Therefore, we concentrate our attention on general scenario with common classification network. This means the units in the output layer are equal to the pre-collected speaker categories in training set.

2.3. Length normalization in common end-to-end deep speaker embedding system

For open-set SV task, since it is impossible to classify testing utterances to known identities in training set, the end-to-end classification network plays role as an automatic speaker embedding extractor, as demonstrated in Fig. 2. Once deep speaker embeddings (e.g. x-vectors) are extracted, just the same as in i-vector approach, cosine similarity or length normalization followed by PLDA is commonly required to get the final pairwise scores. It's noticed that no matter in cosine similarity or PLDA modeling, the length normalization is an extra step performed on the extracted speaker embeddings, and out of end-to-end manner.

3. Deep length normalization

As described in section 2.1, back-end modeling in conventional i-vector approach usually performs on the unit-length hyperspace. When it turns into end-to-end deep neural network, however, in practice the back-end softmax classifier commonly adopts the inner-product based FC layer without normalization. It means that if we want to perform cosine similarity or PLDA on the extracted deep speaker embeddings, such as the representative x-vectors, we should manually normalize them with unit-length first.

It motivates us that whether it is possible to learn the deep speaker embeddings being length-normalized in an end-to-end manner within common classification network. One might wonder the substantial difference between length normalization in an end-to-end manner or out of end-to-end manner. This issue has been studied by [23, 24] in computer vision community. The effect of deep length normalization is equivalent to adding an L_2-constraint on the original loss function. With deep speaker embeddings being length-normalized inherently in an end-to-end manner, our optimization object requires not only the speaker embeddings being separated, but also constrained on a small unit hyperspace. This makes it more difficult to train the network, but in the other side, could greatly enhance its generalization capability.

To this end, a naive practice is just to add an L_2-normalization layer before the output layer. However, we find that the training process may not converge and lead to rather poor performance, especially when the number of output categories is very large. The reason might be that the surface area of the unit-length hypersphere would have not enough room to not only accommodate so many speaker embeddings, but also allow each category of them to be separable.

As done in [23, 24], we introduce a scale parameter α to shape the length-normalized speaker embeddings into suitable radius. The scale layer can scale the unit-length speaker embeddings into a fixed radius given by the parameter α. Therefore, the complete formula of our introduced deep length normalization can be expressed as

$$\boldsymbol{y_i} = \alpha \times \frac{f(\boldsymbol{x}_i)}{\|f(\boldsymbol{x}_i)\|_2} \tag{1}$$

where $\boldsymbol{x_i}$ is the i^{th} input data sequence in the batch, $f(\boldsymbol{x_i})$ is the corresponding output of the penultimate layer of the network, and $\boldsymbol{v_i}$ is the deep normalized embedding.

Our end-to-end system architecture with deep feature normalization is demonstrated in Fig. 3. The length-normalized speaker embedding can be directly fed into the output layer, and all the components in the network are optimized jointly with a unified cross-entropy loss function:

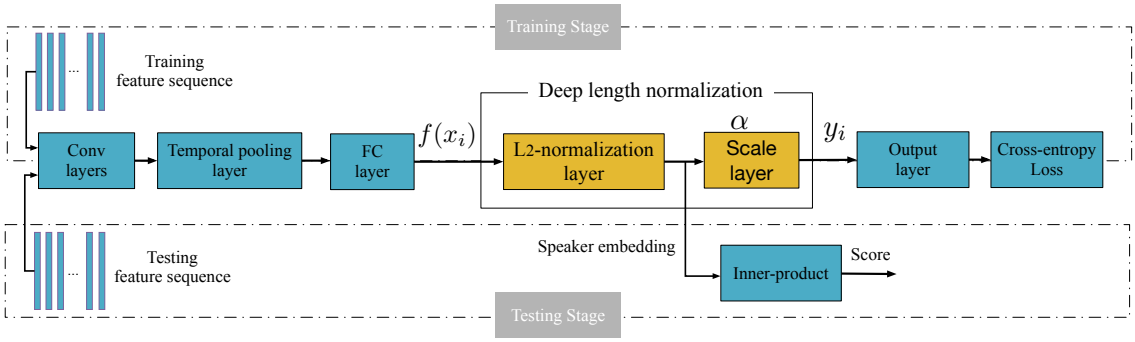


Figure 3: *Deep length normalization in end-to-end speaker verification system: before the final output layer, an L_2-normalization layer followed by a scale layer is added. Therefore, deep speaker embeddings can be inherently length-normalized in an end-to-end manner*

.

$$\ell_s = -\frac{1}{M}\sum_{i=1}^{M}\log\frac{e^{\boldsymbol{W}_{c_i}^T\boldsymbol{y_i}+\boldsymbol{b}_{c_i}}}{\sum_{j=1}^{C}e^{\boldsymbol{W}_j^T\boldsymbol{y_j}+\boldsymbol{b}_j}} \tag{2}$$

where M is the training batch size, C is the output categories, $\boldsymbol{y_i}$ is the deep normalized embedding, c_i is the corresponding ground truth label, and $\boldsymbol{W}$ and $\boldsymbol{b}$ are the weights and bias for the last layer of the network which acts as a back-end classifier.

In total, only a single scalar parameter α is introduced, and it can be inherently trained with other components of the network together. This scale parameter α has a crucial impact on the performance since it determines the radius of the length-normalized hyperspace. The network could have stronger L_2-constraint on the small radius hyperspace with smaller α, but faces the risk of not convergent.

Therefore, it is vital to choose appropriate α and normalize the feature into hyperspace with suitable radius. For elegance, we may prefer to make the parameter α automatically learned by back-propagation. However, because the cross-entropy loss function only takes into account whether it the speaker embeddings are separated correctly, it is apt to increases the value of α to meet the demand. Therefore, the value of α learned by the network might always be high, which results in a relaxed L_2-constraint [23].

A better practice considers α as a hype-parameter, and fix it with a low-value constant in order to enlarge the L_2-constraint. However, too small α for large number of categories may lead to the unconverged case. Hence, we should find an optimal balance point for α.

Given the number of categories C for a training dataset, in order to achieve a classification probability score of p, the authors in [23] give the formulation of theoretical lower bound on α by

$$\alpha_{low} = \log\frac{p(C-2)}{1-p} \tag{3}$$

At the testing stage, speaker embeddings are extracted after the L_2-normalization layer. Since the embeddings have already been normalized to unit length, a simple inner-product or PLDA can be adopted to get the final similarity scores.

4. Experiments

4.1. Data description

Voxceleb1 is a large scale text-independent SR dataset collected "in the wild", which contains over 100,000 utterances from 1251 celebrities [25]. We focus on its open-set verification task.

Table 1: *Baseline end-to-end system architecture*

Layer	Output size	Downsample	Channels	Blocks
Conv1	$64 \times L$	False	16	-
Res1	$64\times L$	False	16	3
Res2	$32 \times \frac{L}{2}$	True	32	4
Res3	$16 \times \frac{L}{4}$	True	64	6
Res4	$8 \times \frac{L}{8}$	True	128	3
Average pool	128	-	-	-
FC (embedding)	128	-	-	-
Output	speaker categories	-	-	-

There are totally 1211 celebrities in the development dataset. The testing dataset contains 4715 utterances from the rest 40 celebrities. There are totally 37720 pairs of trials including 18860 pairs of true trials. To evaluate the system performance, we report results in terms of equal error-rate (EER) and the minimum of the normalized detection cost function (minDCF) at P_{Target} = 0.01 and P_{Target} = 0.001, as shown in Table 2 and Table 3.

4.2. Referenced i-vector system

We build a referenced i-vector system based on the Kaldi toolkit [26]. Firstly, 20-dimensional mel-frequency cepstral coefficients (MFCC) is augmented with their delta and double delta coefficients, making 60-dimensional MFCC feature vectors. Then, a frame-level energy-based voice activity detection (VAD) selects features corresponding to speech frames. A 2048-components full covariance GMM UBM is trained, along with a 400-dimensional i-vector extractor and full rank PLDA.

4.3. End-to-end system

Audio is converted to 64-dimensional log mel-filterbank energies with a frame-length of 25 ms, mean-normalized over a sliding window of up to 3 seconds. A frame-level energy-based voice activity detection (VAD) selects features corresponding to speech frames. In order to get higher level abstract representation, we design a deep convolutional neural network (CNN) based on the well-known ResNet-34 architecture [27], as described in Table 1. Followed by the front-end deep CNN, we adopt the simplest average pooling layer to extract the utterance-level mean statistics. Therefore, given input data sequence of shape $64 \times L$, where L denotes variable-length data frames, we finally get 128-dimensional utterance-level rep-

Table 2: *Voxceleb1 open-set verification task performance, in comparing the effect of our introduced deep length normalization strategy and traditional extra length normalization step (lower is better)*

System Description	Deep L_2-norm	Traditional L_2-norm	Similarity Metric	minDCF10^{-2}	minDCF10^{-3}	$EER(\%)$
i-vector + inner-product	N/A	✗	inner-product	0.736	0.800	13.80
i-vector + cosine	N/A	✓	inner-product	0.681	0.771	13.80
i-vector + PLDA	N/A	✗	PLDA	0.488	0.639	5.48
i-vector + L_2-norm + PLDA	N/A	✓	PLDA	0.484	0.627	5.41
Deep embedding + inner-product	✗	✗	inner-product	0.758	0.888	7.42
Deep embedding+ cosine	✗	✓	inner-product	0.553	0.713	5.48
Deep embedding+ PLDA	✗	✗	PLDA	0.524	0.739	5.90
Deep embedding + L_2-norm + PLDA	✗	✓	PLDA	0.545	0.733	5.21
L_2-normalized deep embedding + inner-product	✓	✗	inner-product	**0.475**	**0.586**	5.01
L_2-normalized deep embedding + PLDA	✓	✗	PLDA	0.525	0.694	**4.74**

Table 3: *Verification performance on VoxCeleb1 for various scale parameter α (lower is better)*

System Description	minDCF10^{-2}	minDCF10^{-3}	$EER(\%)$
Deep embedding baseline	0.553	0.713	5.48
fixed $\alpha = 1$	0.922	0.967	10.18
fixed $\alpha = 4$	0.601	0.828	6.36
fixed $\alpha = 8$	0.515	0.687	5.49
fixed $\alpha = 12$	**0.475**	**0.586**	**5.01**
fixed $\alpha = 16$	0.499	0.596	5.32
fixed $\alpha = 20$	0.503	0.637	5.46
fixed $\alpha = 24$	0.502	0.638	5.54
fixed $\alpha = 28$	0.497	0.640	5.52
trained $\alpha = 26.1$	0.486	0.599	5.60

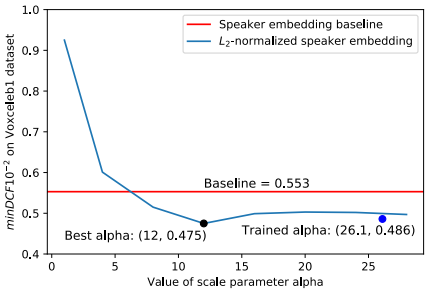
resentation.

The model is trained with a mini-batch size of 128, using typical stochastic gradient descent with momentum 0.9 and weight decay 1e-4. The learning rate is set to 0.1, 0.01, 0.001 and is switched when the training loss plateaus. For each training step, an integer L within [300,800] interval is randomly generated, and each data in the mini-batch is cropped or extended to L frames. After model training finished, the 128-dimensional speaker embeddings are extracted after the penultimate layer of neural network.

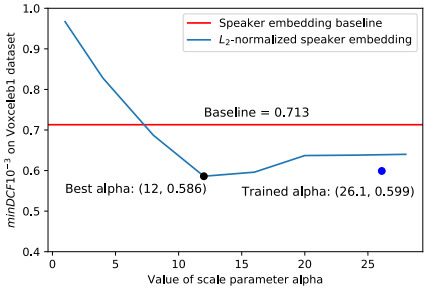
4.4. Evaluation

We first investigate the setting of scale parameter α. For those systems in Table 3 and Fig. 4, the cosine similarity or equivalently L_2-normalized inner-product is adopted to measure the similarities between speaker embeddings. From Fig. 4, we can observe the proposed L_2-normalized deep embedding system achieves the best minDCF of 0.475, 0.586 and EER of 5.01%, which outperforms the baseline system significantly. According to Equation (3), for speaker categories C of 1211 and probability score p of 0.9, the theoretical lower bound of α is 9. The performance is poor when α is below the lower bound and stable with α higher than the lower bound. The best α in our experiment is 12, which is slightly larger than the lower bound.

We further compare the effect of deep length normalization strategy and traditional extra length normalization in the whole SV pipeline. The results are shown in Table 2. No matter in i-vector or baseline deep speaker embedding systems, extra length normalization step followed by PLDA scoring achieves the best performance. When it turns into L_2-normalized deep speaker embedding systems, since the speaker embeddings extracted from the neural network have already been normalized



(a) Performance in terms of minDCF10^{-2}



(b) Performance in terms of minDCF10^{-3}

Figure 4: *Performance tendency curve considering various α*

to unit length, we need no more extra length normalization step. In the testing stage, a simple inner-product achieves the best performance, even slightly better than the PLDA scoring result. It might be the reason that our L_2-normalized speaker embeddings are highly optimized, which could be incompatible with the objective function introduced by PLDA.

5. Conclusions

In this paper, we explore a deep length normalization strategy in end-to-end SV system. We add an L_2-normalization layer followed by a scale layer before the output layer of the deep neural network. This simple yet efficient strategy makes the learned deep speaker embeddings being normalized in an end-to-end manner. The value of scale parameter α is crucial to the system performance especially when the number of output categories is large. Experiments show that system performance could be significantly improved by setting a proper value of α. In the testing stage of an L_2-normalized deep embedding system, a simple inner-product can achieve the state-of-the-art.

6. References

[1] W. Campbell, D. Sturim, and D. Reynolds, "Support vector machines using gmm supervectors for speaker verification," *IEEE Signal Processing Letters*, vol. 13, no. 5, pp. 308–311, 2006.

[2] T. Kinnunen and H. Li, "An overview of text-independent speaker recognition: From features to supervectors," *Speech Communication*, vol. 52, no. 1, pp. 12–40, 2010.

[3] J. H. Hansen and T. Hasan, "Speaker recognition by machines and humans: A tutorial review," *IEEE Signal processing magazine*, vol. 32, no. 6, pp. 74–99, 2015.

[4] D. Reynolds and R. Rose, "Robust text-independent speaker identification using gaussian mixture speaker models," *IEEE Transactions on Speech & Audio Processing*, vol. 3, no. 1, pp. 72–83, 1995.

[5] N. Dehak, P. 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.

[6] D. Reynolds, T. Quatieri, and R. Dunn, "Speaker verification using adapted gaussian mixture models," in *Digital Signal Processing*, 2000, p. 1941.

[7] J. Gonzalez-Dominguez, I. Lopez-Moreno, H. Sak, J. Gonzalez-Rodriguez, and P. J. Moreno, "Automatic language identification using long short-term memory recurrent neural networks," in *Proc. INTERSPEECH 2014*, 2014.

[8] L. Chao, M. Xiaokong, J. Bing, L. Xiangang, Z. Xuewei, L. Xiao, C. Ying, K. Ajay, and Z. Zhenyao, "Deep speaker: an end-to-end neural speaker embedding system," 2017.

[9] 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 *Proc. IEEE SLT 2017*, pp. 165–170.

[10] W. Cai, J. Chen, and M. Li, "Exploring the encoding layer and loss function in end-to-end speaker and language recognition system," in *Proc. Speaker Odyssey*, 2018.

[11] W. Cai, Z. Cai, W. Liu, X. Wang, and M. Li, "Insights into end-to-end learning scheme for language identification," in *Proc. ICASSP 2018*, 2018.

[12] H. Bredin, "Tristounet: triplet loss for speaker turn embedding," in *Proc. ICASSP 2017*, 2017, pp. 5430–5434.

[13] W. Cai, Z. Cai, X. Zhang, and M. Li, "A novel learnable dictionary encoding layer for end-to-end language identification," in *Proc. ICASSP 2018*, 2018.

[14] D. Garcia-Romero and C. Y. Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems." in *Proc. INTERSPEECH*, 2011, pp. 249–252.

[15] D. Snyder, G. Garcia-Romero, D. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *Proc. ICASSP 2018*, 2018.

[16] P. Bousquet, A. Larcher, D. Matrouf, J. Bonastre, and O. Plchot, "Variance-spectra based normalization for i-vector standard and probabilistic linear discriminant analysis," in *Proc. Odyssey*, 2012.

[17] D. Garcia-Romero and A. McCree, "Insights into deep neural networks for speaker recognition," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[18] S. Prince and J. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *ICCV 2007*, pp. 1–8.

[19] P. Kenny, "Bayesian speaker verification with heavy tailed priors," in *Proc. Odyssey Speaker and Language Recogntion Workshop, Brno, Czech Republic*, 2010.

[20] C. Zhang and K. Koishida, "End-to-end text-independent speaker verification with triplet loss on short utterances," in *Proc. Interspeech 2017*, 2017, pp. 1487–1491.

[21] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 815–823.

[22] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "Sphereface: Deep hypersphere embedding for face recognition," in *Proc. CVPR 2017*, vol. 1, 2017.

[23] R. Ranjan, C. D. Castillo, and R. Chellappa, "L2-constrained softmax loss for discriminative face verification," *arXiv preprint arXiv:1703.09507*, 2017.

[24] F. Wang, X. Xiang, J. Cheng, and A. L. Yuille, "Normface: L2 hypersphere embedding for face verification," in *Proceedings of the 25th ACM international conference on Multimedia*. ACM, 2017.

[25] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset," in *Proc. INTERSPEECH 2017,*.

[26] 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*.

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR 2016*, 2016, pp. 770–778.