Combining LSTM, NetVLAD and Additive Margin Softmax for Text-Independent Speaker Verification

Tuomas Kaseva, Mikko Kurimo, *Member, IEEE*

***Abstract*—We propose a deep learning based speaker verification system which combines long-short term memory (LSTM) cells, NetVLAD and additive margin softmax loss. The speaker embedding extraction procedure of the system is based on three steps. First, the input is windowed, then, each window embedded and finally the created embeddings averaged. We abbreviate this method as SEA: split, embed and average, and use it to alleviate the disadvantages of LSTMs when processing very long sequences. We compare our system to x-vector and convolutional neural network motivated speaker verification systems which have demonstrated state-of-the-art performance on the Voxceleb1 dataset at the time they were published. We train our system with the Voxceleb2 dataset which is the same training set as used in the aforementioned systems, and show that our system achieves better results on Voxceleb1. Consequently, we show that unlike in previous work, the additive margin softmax loss gives significant improvements when used with NetVLAD.**

***Index Terms*—additive margin softmax loss, NetVLAD aggregation, recurrent neural network, SEA**

I. Introduction

In recent years, deep learning motivated approaches have shown significant progress in speaker verification. We consider three main reasons for their success. Firstly, larger and more realistic speakers-in-the-wild speaker recognition datasets have become available to the public [1]–[3]. Secondly, the loss functions used in the training of neural networks have advanced. In general, the main objective of the neural networks designed for speaker verification is to transform a given recording into a speaker embedding which embodies the speaker characteristics of the recording [4]–[7]. In the most current methods, the embeddings are learned in a speaker identification process, were original softmax loss is modified by adding a margin to the class decision boundaries [8]–[10] . This allows efficient training and reduces the intra-class variance of the created embeddings [11]–[13].

Finally, the neural network architectures have developed. One of the most prominent discoveries has been x-vectors, speaker embeddings which are extracted from an architecture based on time-delay neural networks (TDNNs) [4], [8], [10]. X-vectors have been shown to outperform i-vectors, which have enjoyed a state-of-the-art status in speaker verification for a long time [14]. In some cases, i-vectors have also been inferior to the speaker verification systems which utilize convolutional neural networks (CNNs) [2], [15]. Furthermore, novel aggregation methods for neural networks have been proposed. Whereas average pooling has been used extensively before, the most recent approaches include statistics pooling, attentive statistics pooling and NetVLAD (vector of locally aggregated descriptors) [9], [16], [17].

In addition, recurrent neural networks (RNNs) with long-short term memory (LSTM) cells [18] have been experimented with [6], [19], [20]. Most importantly, they have shown success in a related task, online speaker diarization [21]–[23]. In this task, LSTMs have been been able to create compact speaker embeddings from very short segments. Although x-vectors have also been used in a similar setting [24], it is questionable whether TDNNs would be the better choice for embedding short audio fragments. In the case of CNNs, the authors of [9] illustrated that the performance of a CNN-based speaker verification system decreased rapidly when the duration of the input segment was shortened.

On the other hand, unlike CNNs and TDNNs, LSTMs are not generally well suited for handling sequences where the number of time steps is in thousands [25]. The problem with the LSTMs is their inability to model long-term memory [25]. For this reason, LSTM-based speaker verification systems usually window the input sequence to smaller segments and embed the segments instead of the whole sequence. The created embeddings are then averaged into a single embedding [19]. Here, we refer to this process as *SEA*: split, embed and average.

In theory, using SEA would allow the neural network to specialize in embedding short audio segments but be effective also with longer recordings. This feature would be advantageous both in speaker verification and speaker diarization. However, it is yet to be proven that LSTM-based speaker verification systems could actually outperform CNN-based systems or x-vectors.

Contributions. We construct a novel speaker verification system that relies on a neural network which integrates LSTMs, NetVLAD and additive margin (AM)-softmax loss [11]. In the system, the speaker embedding extraction is based on the SEA method, and the neural network is trained to embed segments with fixed 2 second duration. We train our system using the Voxceleb2 dataset [2] and evaluate it with the Voxceleb1 dataset [1]. The size of the recordings in Voxceleb1 varies and is generally much longer than 2 seconds. The system is compared to CNN- [9] and x-vector-based speaker verification systems [10] which both are trained with the exact same dataset as ours and have achieved state-of-the-art results on the Voxceleb1 dataset at the time they were published. We demonstrate that our system can exceed the results of both systems. Moreover, we show that on the contrary to the previous results [9], the AM-softmax loss can be very beneficial when combined with NetVLAD.

In addition, we propose a heuristic algorithm which can perform automatic cleaning of a speaker recognition dataset. We apply this algorithm to the Voxceleb2 dataset [2] which is used for training of our neural network. The results illustrate

that the algorithm works and is generally beneficial.

Related work. Our approach has some similarities with Wan et al. [19]. As in their work, we use SEA and LSTMs. However, unlike them, we do not apply generalized end-to-end loss for neural network training. Instead, we use the AM-softmax loss [11]. Furthermore, unlike them, we combine LSTMs with NetVLAD. Although NetVLAD layer has been previously used for speaker verification [9], in that study, the layer was connected to a CNN. NetVLAD has been originally designed for aggregation of CNNs [17] and to the best of our knowledge, we are the first to use it with LSTMs in any application.

II. Proposed methods

In this section, we detail SEA method, illustrated in Fig. 1, and two neural networks developed in this study. The first network uses LSTMs and NetVLAD whereas the second is otherwise the same but replaces NetVLAD with average pooling. The purpose of the second network is to function as a baseline.

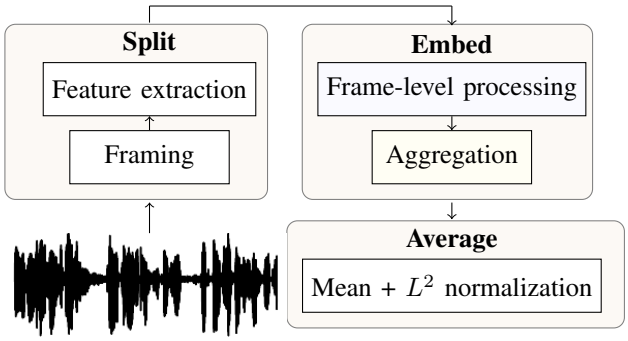

Fig. 1. Schematic of our speaker embedding extraction approach SEA: split, embed and average.

Split. At the beginning, a given audio input is split into overlapping frames with short, roughly 2 seconds or less, duration. Time-varying features are then extracted from each frame, resulting in a set of feature sequences $\mathbf{x}$. The sequences consist of 30 Mel-Frequency Cepstral Coefficients (MFCC) which are extracted every 10ms with 25ms frame length. Thus, the dimensionality of $\mathbf{x}$ is $\mathbb{R}^{30\times T}$, where T is the number of frames. Every $\mathbf{x}$ is also normalized with zero mean and unit variance. The use of MFCCs is motivated based on their success with x-vectors [5], [8], [10].

Embed. In the next step, each $\mathbf{x}$ is transformed into a speaker embedding. In our experiments, we design two neural networks for speaker embedding extraction. The architectures of both networks can be divided to two components: frame-level processing and aggregation. Whereas frame-level processing is performed similarly in both, aggregation is not.

In frame-level processing, each $\mathbf{x}$ is projected into higher level frame-features $\mathbf{h}$. In our approach, $\mathbf{x}$ is fed to a cascade of three bidirectional LSTM layers with skip connections. Each layer outputs the hidden states of both the forward and backward LSTMs. These outputs are concatenated resulting in $\mathbf{h}$ as illustrated in in Figure 2. The structure of the cascade adheres to [23].

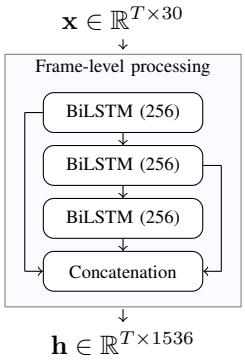

Fig. 2. Frame-level processing. The numbers refer to the number of hidden units in each layer.

In aggregation, the higher level features $\mathbf{h}$ are compressed into a speaker embedding $\mathbf{f}$. The aggregation components are illustrated in Fig. 3. Note that the aggregation component with average pooling has a slightly different configuration than its NetVLAD motivated counterpart. This choice was based on balancing the number of parameters in both neural networks.

We force the embeddings to be L^2 normalized in both components. As a result, cosine distance is the most natural distance metric between different embeddings. A rectified linear unit activation is used in all of the fully connected (FC) layers. We also apply batch normalization [26] after each layer except L^2 normalization layers. This means that the last two layers of both components perform normalization. Although this might seem strange, we discovered it to be beneficial in the preliminary experiments.

The operation of the NetVLAD layer can be summarized as follows. Let us denote the output of the preceding FC layer as $\mathbf{v} \in \mathbb{R}^{T\times 256}$. First, $\mathbf{v}$ is transformed into $\mathbf{V} \in \mathbb{R}^{K\times 256}$ according to a formula [17]

$$\mathbf{V}(k,d) = \sum_{t=1}^{T} \frac{e^{\mathbf{w}_k^T \mathbf{v}_t + b_k}}{\sum_{k'=1}^{K} e^{\mathbf{w}_{k'}^T \mathbf{v}_t + b_{k'}}} (\mathbf{v}_{td} - \mathbf{c}_{kd}), \qquad (1)$$

where $\mathbf{c} \in \mathbb{R}^{K\times 256}$, $\mathbf{w} \in \mathbb{R}^{K\times 256}$ and $\mathbf{b} \in \mathbb{R}^{K}$ are learnable parameters. In this formulation, $\mathbf{c}$ can be interpreted as a set of K cluster centers which characterize the distribution of $\mathbf{v}$ [9]. More specifically, $\mathbf{V}$ consists of first order statistics of residuals $\mathbf{v}_d - \mathbf{c}_k$ in which each element is weighted based on $\mathbf{v}$ and the cluster index k. The number of clusters K is given as an input to the layer. After calculation of the residuals, each row of $\mathbf{V}$ is first L^2 normalized and then concatenated resulting in $\mathbf{V_f} \in \mathbb{R}^{256*K}$. In the literature, additional L^2 normalization operation has been applied after flattening [9], [17]. However, we use batch normalization instead. We found this normalization to perform generally better in the preliminary experiments. The use of NetVLAD in this study is motivated by its recent success in speaker verification when combined with CNNs [9]. Here, we wish to investigate whether NetVLAD could be beneficial also with LSTMs.

Average. In the final stage, we formulate a single embedding $\mathbf{f}_c \in \mathbb{R}^{700}$ for the recording by averaging the created

speaker embeddings and L^2 normalizing the average. In practice, we fit spherical K-means [27] with $K = 1$ to the embeddings and define $\mathbf{f}_c$ as the cluster center of this clustering result. When considering $\mathbf{f}_{c1}$ and $\mathbf{f}_{c2}$ extracted from two different recordings, our system performs speaker verification by computing cosine distance between the embeddings and by thresholding the obtained value. Another popular method for comparing the embeddings is Probabilistic Discriminant Analysis (PLDA) [4], [28]. PLDA could result in performance improvements [8], but since the complexity of our system would then increase, we do not utilize it in this study.

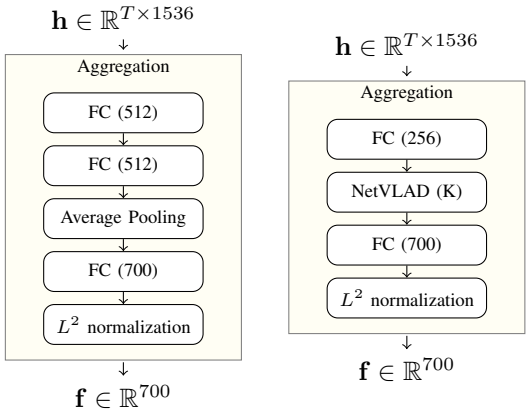

Fig. 3. Two different aggregation approaches: average pooling on the left and NetVLAD on the right. FC refers to a fully connected layer and the numbers to the output dimensionality.

III. Experiments

Data. We use two training sets which are both generated from Voxceleb2 [2]. In the first, abbreviated as $VC2$, all recordings in Voxceleb2 are windowed into 2 second samples with 1 second overlap. The reason for this choice is the training objective of our neural networks that is to identify a speaker from a given training set based on a 2 second segment of speech. The duration was not selected arbitrarily: we experimented also with setting it to 1 and 2.5 seconds. The former was too short for neural networks to learn speaker characteristics properly and the latter did not generally improve the performance of the networks. $VC2$ consists of roughly 6.83 million training samples from 5994 speakers.

The second set, $VC2_C$, is otherwise the same as $VC2$ but excludes a portion of the samples based on a heuristic cleaning algorithm. Given samples S_i belonging to i-th speaker in $VC2$, the algorithm operates in four steps:

1) Create a speaker embedding $\mathbf{f}$ for each sample in S_i.
2) Cluster the embeddings with spherical K-means setting $K = 2$ into groups G_1 and G_2.
3) Calculate the average of silhouette coefficients ϕ of the clustering result. Further details of these coefficients are given in [29].
4) If $|G_1| > 0.6|G_1 \cup G_2|$ and $\phi > 0.3$, exclude all samples belonging to G_2 from the training set. Here, $|G_i|$ refers to a number of elements in group G_i.

In summary, the algorithm investigates whether the recordings initially assigned to a single speaker might contain also another speaker. The algorithm removes samples from S_i only if the speech material portions of the clusters are not balanced and if the clustering result has a high reliability. This reliability is measured using silhouette coefficients. The motivation for this algorithm came from our listening tests which confirmed that Voxceleb2 included wrongly labeled speaker identities in some cases. The exclusions removed approximately 46k samples from $VC2$ but retained the number of speakers, 5994. Speaker embedding extraction was performed using an initial neural network which has the same average pooling based architecture as described in the previous section, but was trained only with 4000 speakers from $VC2$.

We evaluate our models also using the cleaned versions of Voxceleb1 verification test sets, Voxceleb1-test (VC_t), Voxceleb1-H (VC_H) and Voxceleb1-E (VC_E) [2]. The recordings in these sets are framed to 2 second duration segments with 1.5 seconds overlap. The overlap duration was determined in the preliminary experiments.

We construct also our own verification set from the development set of Voxceleb1. This set is used for model evaluation during training. The set consists of speech segments with a fixed 2 seconds duration, and which each are extracted from a unique session and speaker. The number of extracted segments is about 20k and they belong to 1211 speakers. We form close to 150k segment pairs where half of the pairs correspond to the same speaker and the other half to different speakers. We name this verification set as VC_{2sec}.

Training. In training, the output of the aggregation component is connected to a fully connected layer which is used for a speaker identification task. This is illustrated in Fig. 4. Training has two stages: warm-up with the softmax loss and fine-tuning with the AM-softmax loss [11]. In the warm-up, the neural network is trained for 5 epochs, using Adam optimizer with 0.01 learning rate. Batch size is chosen as 512. We generally observed that the performance of the neural networks on the VC_{2sec} would not improve after the fifth epoch when using the softmax loss.

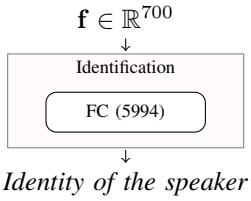

Fig. 4. Identification with 5994 different speakers from Voxceleb2.

In the fine-tuning, the softmax loss for i-th training sample is reformulated as

$$L_i = log \frac{e^{s(\mathbf{W}_{y_i}^T \mathbf{f} - m)}}{e^{s(\mathbf{W}_{y_i}^T \mathbf{f} - m)} + \sum_{j=1, j\neq y_i}^{5994} e^{s\mathbf{W}_j^T \mathbf{f}}}, \quad (2)$$

where y_i is the label of i-th training sample, $\mathbf{W} \in \mathbb{R}^{700\times 5994}$ a learnable weight matrix with all rows L^2 normalized and s and m a given scale and margin. Equation 2 is known as the AM-softmax loss [11]. We set $m = 0.15$ and $s = 0.25$ based on our preliminary experiments. $\mathbf{W}$ is initialized with

the weights of the best neural network configuration found in the warm-up.

The main point of using the AM-softmax loss is to decrease intra-class variance, which is generally difficult with the softmax loss [11]–[13]. In other words, the higher the margin m is set, the more closer, in terms of cosine distance, the speaker embeddings belonging to the same class are forced. The cosine distance metric arises from the L^2 normalizations of both $\mathbf{f}$ and the rows of $\mathbf{W}$. The scale of s is generally set to a some high value to ensure convergence [30]. In recent years, the AM-softmax loss and other similar methods [12], [13] have emerged as state-of-the-art approaches in speaker verification [8]–[10], [31].

The fine tuning is continued for 10 epochs with otherwise the same setting as in warm-up. We monitor the progress of the training by first computing cosine distances between the embeddings of each pair in VC_{2sec} and then calculating equal error rate (EER) on these distances after each epoch. EER is a standard error metric in speaker verification [2], [4], [9]. Although the VC_{2sec} contains over 150k pairs, the evaluation on this set is efficient during the training since it consists of short, equal length segments which can be embedded rapidly. We save the weights of the neural network after each epoch, and choose the configuration with the best EER value as our final model.

IV. Results

In this section, we first investigate the effect of the cleaning algorithm, aggregation and the AM-softmax loss. Finally, we present a results comparison. We use EER as an evaluation metric in all experiments.

TABLE I
EFFECT OF TRAINING SET (EER %). $K = 30$.

Aggregation	Training set	VC_t	VC_E	VC_H	VC_{2sec}
NetVLAD	$VC2$	2.49	2.47	4.53	**6.65**
NetVLAD	$VC2_C$	**2.18**	**2.45**	**4.45**	6.66

Effect of dataset cleaning. In Table I, we show that marginal improvements can be achieved with the use of our data cleaning algorithm. This proves that the algorithm is reasonable and also encourages discussion whether some cleaning operation should be applied to Voxceleb2. However, the improvements in every other test sets except VC_t are minor and with VC_{2sec}, the cleaning has not been beneficial.

TABLE II
EFFECT OF K AND AGGREGATION (EER %). TRAINING SET = $VC2_C$.

Aggregation	K	VC_t	VC_E	VC_H	VC_{2sec}
Average pooling	-	2.46	2.45	4.42	7.05
NetVLAD	8	2.41	2.40	**4.35**	6.92
NetVLAD	14	2.32	**2.37**	4.36	6.68
NetVLAD	30	**2.18**	2.45	4.45	**6.66**

Effect of aggregation approach. Table II investigates the performance of the two aggregation approaches and the choice of K. The results show that NetVLAD is the better approach. This is particularly clear with VC_{2sec}. However, the best scores with different test sets are all obtained with different K values. This result highlights the importance of using multiple different test sets for model evaluation. Nevertheless, we can decide on the best model based on the average over all EER scores. In this case, the NetVLAD-based aggregation with $K = 14$ has the best performance.

Effect of loss function. Table III illustrates that the AM-softmax loss brings significant improvements over the softmax loss. Similar results were obtained with the average pooling aggregation. However, we want to emphasize the results with the NetVLAD aggregation since in [9], the use of NetVLAD with the AM-softmax loss has not resulted in notable performance improvements. Here, we demonstrate that the two can be combined successfully. The results with different K values were essentially the same.

TABLE III
EFFECT OF LOSS FUNCTION (EER %). $K = 30$, TRAINING SET = $VC2_C$.

Aggregation	Loss	VC_t	VC_E	VC_H	VC_{2sec}
NetVLAD	Softmax	3.25	3.30	5.90	8.40
NetVLAD	AM-softmax	**2.18**	**2.45**	**4.45**	**6.66**

Results comparison. In Table IV, we compare our system to a x-vector and [10] and a CNN-based [9] speaker verification systems. Our system uses NetVLAD aggregation component with $K = 14$. The comparison is fair since all the systems are trained with the same dataset, Voxceleb2, and because the number of parameters are close to each other: 4.2 million in [10], 7.7 million in [9] and 6.7 million in our system. Nevertheless, we want to address that the current state-of-the-art results on each Voxceleb1 test set have been achieved by the top teams of the VoxCeleb Speaker Recognition Challenge 2019. However, the comparison with the best systems would not be reasonable here since they all use data augmentation and have a lot more parameters [32].

Our system exceeds the scores obtained by [9] with a comfortable margin. When comparing with [10], the differences are not that large but their system also utilizes PLDA. Since we use only cosine distance, our system is much simpler.

TABLE IV
RESULTS COMPARISON (EER %).

System	Scoring	VC_t	VC_E	VC_H
Xie *et al.* [9]	Cosine	3.22	3.13	5.06
Xiang *et al.* [10]	PLDA	2.69	2.76	4.73
Ours	Cosine	**2.32**	**2.37**	**4.36**

V. Conclusion

We have presented a speaker verification system which is based on using a novel neural network. This neural network consists of a cascade of LSTM layers, NetVLAD aggregation layer and uses the AM-softmax loss in training. We have demonstrated that the system achieves promising results with the Voxceleb1 dataset, and that combining NetVLAD with the AM-softmax loss can lead to notable performance improvements. The system is available online[1]. The next steps in the development of the system would consist of speaker diarization experiments, studying augmentation methods and further experimenting with different aggregation approaches.

Acknowledgment

Computational resources were provided by the Aalto Science-IT project.

[1]https://github.com/Livefull/LSTM-based-speaker-verification

REFERENCES

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

[2] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep speaker recognition," in *INTERSPEECH*, 2018.

[3] M. McLaren, L. Ferrer, D. Castan, and A. Lawson, "The speakers in the wild (SITW) speaker recognition database.," in *Interspeech*, pp. 818–822, 2016.

[4] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5329–5333, IEEE, 2018.

[5] D. Snyder, D. Garcia-Romero, G. Sell, A. McCree, D. Povey, and S. Khudanpur, "Speaker recognition for multi-speaker conversations using x-vectors," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5796–5800, IEEE, 2019.

[6] H. Bredin, "Tristounet: triplet loss for speaker turn embedding," in *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 5430–5434, IEEE, 2017.

[7] 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.

[8] Y. Liu, L. He, and J. Liu, "Large margin softmax loss for speaker verification," *arXiv preprint arXiv:1904.03479*, 2019.

[9] W. Xie, A. Nagrani, J. S. Chung, and A. Zisserman, "Utterance-level aggregation for speaker recognition in the wild," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5791–5795, IEEE, 2019.

[10] X. Xiang, S. Wang, H. Huang, Y. Qian, and K. Yu, "Margin matters: Towards more discriminative deep neural network embeddings for speaker recognition," *arXiv preprint arXiv:1906.07317*, 2019.

[11] F. Wang, J. Cheng, W. Liu, and H. Liu, "Additive margin softmax for face verification," *IEEE Signal Processing Letters*, vol. 25, no. 7, pp. 926–930, 2018.

[12] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4690–4699, 2019.

[13] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "Sphereface: Deep hypersphere embedding for face recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 212–220, 2017.

[14] 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, 2010.

[15] M. Ravanelli and Y. Bengio, "Speaker recognition from raw waveform with sincnet," in *2018 IEEE Spoken Language Technology Workshop (SLT)*, pp. 1021–1028, IEEE, 2018.

[16] K. Okabe, T. Koshinaka, and K. Shinoda, "Attentive statistics pooling for deep speaker embedding," *arXiv preprint arXiv:1803.10963*, 2018.

[17] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5297–5307, 2016.

[18] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[19] L. Wan, Q. Wang, A. Papir, and I. L. Moreno, "Generalized end-to-end loss for speaker verification," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4879–4883, IEEE, 2018.

[20] 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)*, pp. 5115–5119, IEEE, 2016.

[21] Q. Wang, C. Downey, L. Wan, P. A. Mansfield, and I. L. Moreno, "Speaker diarization with lstm," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5239–5243, IEEE, 2018.

[22] A. Zhang, Q. Wang, Z. Zhu, J. Paisley, and C. Wang, "Fully supervised speaker diarization," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6301–6305, IEEE, 2019.

[23] G. Wisniewksi, H. Bredin, G. Gelly, and C. Barras, "Combining speaker turn embedding and incremental structure prediction for low-latency speaker diarization," in *Proc. Interspeech*, 2017.

[24] G. Sell, D. Snyder, A. McCree, D. Garcia-Romero, J. Villalba, M. Maciejewski, V. Manohar, N. Dehak, D. Povey, S. Watanabe, *et al.*, "Diarization is hard: Some experiences and lessons learned for the JHU team in the inaugural DIHARD challenge.," in *Interspeech*, pp. 2808–2812, 2018.

[25] S. Li, W. Li, C. Cook, C. Zhu, and Y. Gao, "Independently recurrent neural network (indrnn): Building a longer and deeper RNN," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5457–5466, 2018.

[26] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning*, vol. 37 of *Proceedings of Machine Learning Research*, pp. 448–456, 2015.

[27] S. Zhong, "Efficient online spherical k-means clustering," in *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, vol. 5, pp. 3180–3185, IEEE, 2005.

[28] S. Ioffe, "Probabilistic linear discriminant analysis," in *European Conference on Computer Vision*, pp. 531–542, Springer, 2006.

[29] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.

[30] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu, "Cosface: Large margin cosine loss for deep face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5265–5274, 2018.

[31] Y. Li, F. Gao, Z. Ou, and J. Sun, "Angular softmax loss for end-to-end speaker verification," in *2018 11th International Symposium on Chinese Spoken Language Processing (ISCSLP)*, pp. 190–194, IEEE, 2018.

[32] "Voxceleb speaker recognition challenge 2019." http://www.robots.ox.ac.uk/~vgg/data/voxceleb/competition.html. Accessed: 2019-10-2.