Improving Aggregation and Loss Function for Better Embedding Learning in End-to-End Speaker Verification System

Zhifu Gao\(^1\), Yan Song\(^1\), Ian McLoughlin\(^2\), Pengcheng Li\(^1\), Yiheng Jiang\(^1\), Lirong Dai\(^1\)

\(^1\)University of Science and Technology of China, Hefei, China
\(^2\)School of Computing, University of Kent, Medway, UK

Abstract

Deep embedding learning based speaker verification (SV) methods have recently achieved significant performance improvement over traditional i-vector systems, especially for short duration utterances. Embedding learning commonly consists of three components: frame-level feature processing, utterance-level embedding learning, and loss function to discriminate between speakers. For the learned embeddings, a back-end model (i.e., Linear Discriminant Analysis followed by Probabilistic Linear Discriminative Analysis (LDA-PLDA)) is generally applied as a similarity measure. In this paper, we propose to further improve the effectiveness of deep embedding learning methods in the following components: (1) A multi-stage aggregation strategy, exploited to hierarchically fuse time-frequency context information for effective frame-level feature processing. (2) A discriminant analysis loss is designed for end-to-end training, which aims to explicitly learn the discriminative embeddings, i.e. with small intra-speaker and large inter-speaker variances. To evaluate the effectiveness of the proposed improvements, we conduct extensive experiments on the VoxCeleb1 dataset. The results outperform state-of-the-art systems by a significant margin. It is also worth noting that the results are obtained using a simple cosine metric instead of the more complex LDA-PLDA backend scoring.

Index Terms: speaker verification, speaker embedding, multi-stage aggregation, discriminant analysis loss

1. Introduction

Speaker recognition (SR) is the task of automatically retrieving identity information from a given speech utterance. Generally, it can be categorized into speaker identification (SID) and speaker verification (SV), according to the recognition settings. The former classifies an utterance into a specific identity from a known speaker set, while the latter determines whether the claimed identity of a speaker matches a given enrolment.

Compared to SID, SV is an open-set recognition problem, which means there is no overlap in speakers between the training and test set. In essence, this means that SV is closely related to the metric learning problem, where the key is to learn effective utterance-level representations with small intra-class and large inter-class variances.

In recent years, more attention has been paid to deep learning methods for SV, where deep neural networks (DNNs) are employed to extract speaker representations. Being able to benefit from a discriminative training process, deep embedding methods such as d-vector or x-vector have been shown to outperform traditional i-vectors [1, 2], especially for short duration utterances. Existing deep embedding learning architectures include time-delay DNN (TDNN) [2], convolutional neural network (CNN) [3, 4], and Long Short-Term Memory Network (LSTM) [5]. They generally consist of three main components [6, 7]: (1) Frame-level feature processing to model local short spans of acoustic features via TDNN or convolutional layers. (2) Utterance-level embedding learning, containing a pooling method to map variable-length frame-level features into fixed-length utterance-level representations. (3) Loss function to discriminate directly between speakers. An LDA-PLDA back-end has proved to be crucial to improve performance of i-vector based SV systems, since it can effectively compensate for channel differences [8, 9, 10, 11]. It is also widely used in deep embedding based systems as a back-end model [2, 4, 12].

Many recent works have focused on utterance-level embedding learning, e.g., average pooling [1], statistical pooling [2], attentive pooling [13, 14], cross-convolutional-layer pooling [5], learnable dictionary encoding (LDE) [12]. Besides cross entropy loss (CE), different loss functions have been recently proposed, including triplet loss [15, 16], center loss [12, 17], angular softmax (A-softmax) [12, 18], additive margin softmax (AM-softmax) [19] and logistic margin (LM) [19]. However, it is still challenging to incorporate the effective LDA-PLDA backend into a deep embedding learning architecture. Furthermore, few works have considered frame-level processing [7]. In this paper, we focus on frame-level processing and the loss function for more effective embedding learning while, for utterance-level learning, we employ a statistical pooling method [2]. The overall framework is shown in Fig.1, and will be described in more detail in Section 2.

For frame-level processing, a multi-stage aggregation\(^1\) (MSA) strategy is proposed to exploit hierarchical time-frequency context information. Each stage contains a sequence of convolutional layers, and outputs the feature maps with different channels and time-frequency resolutions. In MSA, the outputs of stages are first convoluted to match time-frequency resolutions, then incorporated into embeddings. This differs from existing frame-level processing, in which the learned features are generally over a single scale.

In terms of loss function, a discriminant analysis loss (DALoss) is proposed to overcome the shortcoming that LDA-PLDA could not be jointly trained with the embedding learning network. This is motivated by work in computer vision [20, 21]. The LDA-PLDA backend cannot be jointly trained with the embedding learning model in current SV systems, but our proposed DALoss can work end-to-end, allowing LDA-PLDA to be jointly trained with CE loss.

To evaluate the effectiveness of the proposed MSA and DALoss, extensive experiments have been conducted. Results show the proposed method obtains 17% relative improvement in terms of equal error rate (EER) over the baseline. To our

\(^1\)Here, the aggregation is defined as the combination of different stages throughout the network instead of the pooling operation itself.
knowledge, this also outperforms other state-of-the-art systems, as detailed in Sec. 4.2.

2. Overview of network architecture

The proposed end-to-end deep embedding learning architecture is shown in Fig. 1. It consists of frame-level processing, utterance-level embedding learning, and loss functions.

The frame-level processing part could be divided into 5 stages according to their time-frequency resolutions, as annotated in the figure, where each stage consists of a sequence of convolutional layers. Stages 0 and 1 have the same time-frequency resolution (e.g. $N \times 40 \times 1$). The resolution of feature maps then halves in both time and frequency axes from stage 2 to 4, generating hierarchical features with pyramid scales. Viewing Fig. 1 from left to right, resolutions change from fine to coarse, corresponding to hierarchical feature context information from local to global [22]. To effectively utilize the hierarchical features from different stages for frame-level feature processing, an MSA strategy is designed which consists of a convolutional layer at each stage to match the time-frequency resolution, and a concatenation operation to fuse them.

A pooling layer then follows the frame-level part to map frame-level features into utterance-level representations. In this paper, we use statistical pooling [2], although it is easy to extend to the other pooling methods mentioned in Section 1. A fully connected (FC) layer, termed an embedding layer, is inserted to make a nonlinear transformation of speaker representations.

The outputs of the FC are firstly length normalized by L2 and then multiplied by a constant scale [4], before being fed into CE and DALoss. To overcome the shortcoming that LDA-PLDA could not be jointly trained with the embedding learning network, we have designed DALoss to perform like LDA-PLDA in an end-to-end network, which could be jointly trained with CE loss. The DALoss is expected to imitate the LDA-PLDA backend in two ways. On one hand, it reduces intra-speaker variances, to make embeddings for each speaker identity more compact. On the other hand, it enlarges inter-speaker variances, to make embeddings for each speaker identity more discriminative. Though the transition $H_l$ can be based on any layers, for efficiency and simplicity, we choose a single convolution with kernel size of 1x1 and stride of 1 followed by batch normalization [25] and a nonlinearity. They are then concatenated and fed into a nonlinear transition layer, formulated as:

$$\Gamma(X) = H_l([x_1^2, ..., x_{M_l}^2]; x_1^1, ..., x_{M_l}^1, x_1^3, ..., x_{M_l}^3]$$

3. Methods

3.1. Multi-stage aggregation (MSA)

Although it is feasible to aggregate any stages throughout the network, we are concerned with aggregating the outputs of stages with different resolutions. We design an MSA strategy to utilize multiple inputs with different resolutions and incorporate them into the outputs, as depicted in Fig. 1.

Specially, we aggregate the outputs from stage 2 to stage 4. We note feature maps of the outputs of the $l$-th stage as,$[x_1^l, x_2^l, ..., x_{M_l}^l]$, $l = 2, 3, 4$, where $M_l$ denotes the number of channels. Since they have different feature map sizes, we utilize convolutions to make them match in time-frequency resolution, as formulated as:

$$[\tilde{x}_1^l, \tilde{x}_2^l, ..., \tilde{x}_{M_l}^l] = Conv(x_1^l, x_2^l, ..., x_{M_l}^l)$$

where $Conv$ could be a single convolution with stride of 2 to downsample the feature maps. It could also be a single transposed convolution or bilinear interpolation to upsample. They are then concatenated and fed into a nonlinear transition layer, formulated as:

$$\Gamma(X) = H_l([x_1^2, ..., x_{M_l}^2]; x_1^1, ..., x_{M_l}^1, x_1^3, ..., x_{M_l}^3])$$

3.2. Discriminant analysis loss (DALoss)

In this subsection, we formulate the DALoss mathematically. We suppose that there are $K$ identities and each identity comprises $N$ utterances in a min-batch. $x_n^k$ denotes the $n$-th embedding of the $k$-th identity. DALoss could be formulated as:

$$L_{DALoss} = \beta S_{intra} + \gamma S_{inter}$$

where $S_{intra}$ denotes the intra-speaker variabilities and $S_{inter}$ represents the inter-speaker variabilities. $\beta$ and $\gamma$ are their loss weights of each respective variance.

Since $S_{intra}$ penalizes the maximum variabilities within each speaker, it could be formulated as:

$$S_{intra} = \sum_{k=1}^{K} S_{intra}^k = \sum_{k=1}^{K} \frac{C_k}{\sum_{j=1}^{C_k} 1/f_j(x_n^m, x_n^k)}$$

3.2. Discriminant analysis loss (DALoss)

In this subsection, we formulate the DALoss mathematically. We suppose that there are $K$ identities and each identity comprises $N$ utterances in a min-batch. $x_n^k$ denotes the $n$-th embedding of the $k$-th identity. DALoss could be formulated as:

$$L_{DALoss} = \beta S_{intra} + \gamma S_{inter}$$

where $S_{intra}$ denotes the intra-speaker variabilities and $S_{inter}$ represents the inter-speaker variabilities. $\beta$ and $\gamma$ are their loss weights of each respective variance.

Since $S_{intra}$ penalizes the maximum variabilities within each speaker, it could be formulated as:

$$S_{intra} = \sum_{k=1}^{K} S_{intra}^k = \sum_{k=1}^{K} \frac{C_k}{\sum_{j=1}^{C_k} 1/f_j(x_n^m, x_n^k)}$$
where \( f_j(x^n, y^n) \) denotes the \( j \)-th largest distance between embeddings of the \( k \)-th identity. The overall cost is the mean of the first \( C_k \) largest distances within each identity. The \( S_{inter} \) loss is designed to compress the distance of those hard samples of the same identity and thus reduce the largest intra-speaker variables.

Then \( S_{inter} \) represents the minimum variabilities between speakers, formulated as:

\[
S_{inter} = \max(0, m - \min(f(\tilde{x}^l, \tilde{x}^l)))
\]

(5)

\[
\tilde{x}^l = \frac{1}{N^l} \sum_{n=1}^{N^l} x^n_i
\]

(6)

where \( \tilde{x}^l \) denotes the center of the \( l \)-th identity in current mini-batch. \( m \) denotes a super parameter of the minimum margin between identity centers. \( N^l \) is the number of embedding of the \( l \)-th identity. \( i \neq j \in [1, ..., K] \). The model pulls the distances between centers of different identities to be larger than the minimum margin by reducing \( S_{inter} \) loss.

We define two kinds of distance metric, formulated as:

\[
f_E(x, y) = \| x - y \|^2
\]

(7)

\[
f_C(x, y) = 1 - \frac{\langle x, y \rangle}{\| x \|_2 \| y \|_2}
\]

(8)

where \( f_E(x, y) \) denotes Euclidean distance and \( f_C(x, y) \) denotes cosine distance, termed DALoss-E and DALoss-C respectively in the following experiments.

4. Experiments

4.1. Experiment setup

We evaluate performance on VoxCeleb1 without data augmentation. The audio is converted to 41-dimensional filter bank outputs (FBank), with a frame-length of 25 ms, and mean-normalized over a sliding window of 3 s. These FBank features are randomly truncated into short slices ranging from 2 s to 4 s, finally generating 3200 input slices per speaker. The network optimizer is stochastic gradient descent (SGD) with a momentum rate of 0.9. The GPU platform is a single GTX1080Ti card and, limited by the GPU memory, we set \( K = 50, C_k = 2, N = 2 \) in a mini-batch. The learning rate is initialized to 0.1, and multiplied by 0.1 every epoch. \( \beta, \gamma \) is set to 0.1 and \( m = 0.2 \) in this paper.

To evaluate the effectiveness of MSA and DALoss, We utilize ResNet [26] and ResNetXi [27] as the network backbone respectively, detailed in Table 2. In this case, stage 0 consists of a single convolution layer with kernel size of 7x7, stride of 1 and padding of 3. Batch normalization and ReLu are added. In the following experiments, when the system name begins with the “R” or “X”, it means the backbone was ResNet or ResNetXt respectively.

<table>
<thead>
<tr>
<th>Frame-level model</th>
<th>Pooling method</th>
<th>Loss function</th>
<th>Similarity</th>
<th>EER (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>I-vectors [23]</td>
<td>-</td>
<td>-</td>
<td>PLDA</td>
<td>8.8</td>
</tr>
<tr>
<td>X-vectors [24]</td>
<td>Statistical pooling</td>
<td>CE</td>
<td>PLDA</td>
<td>7.1</td>
</tr>
<tr>
<td>VGG-M [23]</td>
<td>Average pooling</td>
<td>Contrastive</td>
<td>Cosine</td>
<td>7.8</td>
</tr>
<tr>
<td>ResNet-34 [12]</td>
<td>LDE</td>
<td>Center loss</td>
<td>A-softmax</td>
<td>4.87</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>PLDA</td>
<td>4.48</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>AM-softmax</td>
<td>4.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Cosine</td>
<td>4.29</td>
</tr>
<tr>
<td>R-MSA_3_4</td>
<td>Statistical pooling</td>
<td>DALoss-E</td>
<td>Cosine</td>
<td>4.12</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>DALoss-C</td>
<td>4.09</td>
</tr>
<tr>
<td>X-MSA_3_4</td>
<td>Statistical pooling</td>
<td>DALoss-E</td>
<td>Cosine</td>
<td>3.95</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>DALoss-C</td>
<td>3.87</td>
</tr>
</tbody>
</table>

4.2. Comparison with state-of-the-art systems

We have compared our proposed systems with current state-of-the-art systems, as shown in Table 1. All these systems were trained on VoxCeleb1 [23], without using data augmentation. R-MSA_3_4 and X-MSA_3_4 are our proposed systems, which are combined with MSA and DALoss. In both of them, the MSA module aggregates the outputs of stage 3 and stage 4. The DALoss-E and DALoss-C in the loss column, denotes that the DALoss distance is measured by Euclidean or cosine distance respectively.

Results reveal that the performance of DALoss-C is sightly superior to DALoss-E. It may because there exists a mismatch between training and evaluation criterions in DALoss-E. Under the DALoss-C, R-MSA_3_4 and X-MSA_3_4 obtain 11% and 17% relative improvements over baseline R and X without MSA and DALoss respectively, described in Section 4.3.

The i-vector and x-vector systems are two widely used baselines. Cai et al. [12] investigated the LDE pooling method and discriminative loss, e.g., center loss and A-softmax. LM was proposed for building an end-to-end system in [19]. These systems all obtained large improvements in performance over the baselines. Our proposed systems which incorporate MSA and DALoss have all achieved performance comparable with state-of-the-art. It is worth noting that our systems adopt a simple cosine metric as a similarity measure.
4.3. Evaluation of MSA

In this subsection, we evaluate the performance of MSA separately, as described in Table 3. The LDA-PLDA backend is applied to calculate similarity scores, and we do not add DALoss in this subsection. “DCF” in Table 3, denotes the minimum of the normalized detection cost function at \( P_{\text{target}} = 0.01 \text{ (minDCF)} \). The configurations for comparison are as follows:

- **R**: This is a baseline without MSA, where the backbone network is ResNet-34 [26], as detailed in Table 2.
- **R-MSA_3.4**: These use our proposed MSA method, where we adopt ResNet as backbone. In R-MSA_3.4, the MSA module aggregates the outputs of stage 3 and stage 4. In R-MSA_2.3.4, the MSA module aggregates the outputs of stage 2, stage 3 and stage 4.
- **X**: This is another baseline without MSA, where the backbone network is ResNetXt-41 [27], as detailed in Table 2.
- **X-MSA_3.4**: These also use our proposed MSA method, where ResNetXt is adopted as backbone. In X-MSA_2.3.4, the MSA module aggregates the outputs of stage 3 and stage 4. In X-MSA_2.3.4, the MSA module aggregates the outputs of stage 2, stage 3 and stage 4.

<table>
<thead>
<tr>
<th>System</th>
<th>Cosine EER</th>
<th>LDA-PLDA EER</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>4.92</td>
<td>0.4643</td>
<td>2M</td>
</tr>
<tr>
<td>R-MSA_3.4</td>
<td>4.41</td>
<td>0.4645</td>
<td>2.1M</td>
</tr>
<tr>
<td>R-MSA_2.3.4</td>
<td>4.53</td>
<td>0.4681</td>
<td>2.3M</td>
</tr>
<tr>
<td>X</td>
<td>4.69</td>
<td>0.4397</td>
<td>1.2M</td>
</tr>
<tr>
<td>X-MSA_3.4</td>
<td>4.31</td>
<td>0.3956</td>
<td>1.4M</td>
</tr>
<tr>
<td>X-MSA_2.3.4</td>
<td>4.31</td>
<td>0.4058</td>
<td>1.8M</td>
</tr>
</tbody>
</table>

From Table 3, we can see that the DALoss method is able to improve performance, and even outperform the LDA-PLDA backend. The results confirm that DALoss can effectively reduce intra-speaker variances and enlarge inter-speaker variances, leading to the ability to generate more discriminative speaker embeddings.

4.4. Evaluation of DALoss

In this subsection, we evaluate the performance of DALoss separately, as described in Table 4. For the sake of clarity, we only show the performance of R-MSA_3.4 described in subsection 4.3, since the DALoss is independent of specific embedding learning networks. DALoss is jointly trained with CE loss and we omit CE for short in Table 4.

<table>
<thead>
<tr>
<th>System</th>
<th>Loss</th>
<th>Metric</th>
<th>EER</th>
<th>DCF</th>
</tr>
</thead>
<tbody>
<tr>
<td>R-MSA_3.4</td>
<td>CE</td>
<td>Cosine/PPLDA</td>
<td>4.41</td>
<td>0.4656</td>
</tr>
<tr>
<td></td>
<td>DALoss-E</td>
<td>Cosine</td>
<td>4.25</td>
<td>0.4587</td>
</tr>
<tr>
<td></td>
<td>DALoss-C</td>
<td>Cosine</td>
<td>4.12</td>
<td>0.4570</td>
</tr>
</tbody>
</table>

5. Conclusions

This paper has proposed an improved end-to-end deep embedding learning system for SV. To further enhance performance of deep embedding learning methods, we have proposed MSA and DALoss. The MSA is exploited to incorporate hierarchical features of pyramidal scales and resolutions into speaker embeddings. The results show that MSA is an effective strategy to fuse these multiple complementary features. We have also proposed DALoss to learn more discriminative embeddings with smaller intra-speaker variances and larger inter-speaker variances. It is inherently built in an end-to-end fashion to create a network which can be jointly trained using CE loss, unlike that using LDA-PLDA. Extensive experiments have been conducted on VoxCeleb1. The results show the proposed method obtained 17% relative improvement over baseline to achieve state-of-the-art performance.

Since there are some hyper-parameters in DALoss, it may need some skills to set their values for different conditions. In future, we would extend DALoss to other conditions to study the dependencies between the setups of hyper-parameters and datasets. Moreover, we hope to reduce the number of hyper-parameters.

6. Acknowledgements

The work was supported by National Natural Science Foundation of China grant no U1613211.
7. References


