On the Usage of Multi-feature Integration for Speaker Verification and Language Identification

Zheng Li[1]*, Miao Zhao*[2]*, Jing Li*[2]*, Lin Li*[1]*, Qingyang Hong*[2]

[1]School of Electronic Science and Engineering, Xiamen University, China
[2]School of Informatics, Xiamen University, China

lilin@xmu.edu.cn, qyhong@xmu.edu.cn

Abstract

In this paper, we study the technology of multiple acoustic feature integration for the applications of Automatic Speaker Verification (ASV) and Language Identification (LID). In contrast to score level fusion, a common method for integrating subsystems built upon various acoustic features, we explore a new integration strategy, which integrates multiple acoustic features based on the x-vector framework. The frame level, statistics pooling level, segment level, and embedding level integrations are investigated in this study. Our results indicate that frame level integration of multiple acoustic features achieves the best performance in both speaker and language recognition tasks, and the multi-feature integration strategy can be generalized in both classification tasks. Furthermore, we introduce a time-restricted attention mechanism into the frame level integration structure to further improve the performance of multi-feature integration. The experiments are conducted on VoxCeleb 1 for ASV and AP-OLR-17 for LID, and we achieve 28% and 19% relative improvement in terms of Equal Error Rate (EER) in ASV and LID tasks, respectively.

Index Terms: feature integration, acoustic features, attentive learning, speaker verification, language identification, x-vector

1. Introduction

In the last two decades, significant progress has been made in the fields of Automatic Speaker Verification (ASV) and Language Identification (LID). ASV tasks and LID tasks may both be categorized as classification tasks, even though the implied information in each task is different. Thus, they share some basic speech classification technologies. In the early years, the mainstream technologies of ASV and LID were based on Gaussian mixed models, such as GMM-UBM [1], joint factor analysis (JFA) [2], and the dominant i-vector [3]. With the development of the deep neural network, DNN i-vecor [4], d-vector [5] and x-vector [6] had been proposed in the literature.

Although a series of speaker or language modeling methods have been proposed, most of them utilize only one kind of acoustic feature as the encoder's input. However, due to the differences in extraction algorithms, different kinds of acoustic features may capture unique discriminative information. For example, MFCC features [7] and FBank features share the same Mel filter banks, but MFCC eliminates data redundancy via the Discrete Cosine Transform (DCT); MFCC feature and PLP feature [8] adopt different filter banks.

Nowadays, it is common to fuse several subsystems, which are built on various acoustic features, at the score level to improve the performance [9, 10]. In practical usage, equal weight fusion is a common approach, while some score-level fusion toolkits are available and popular in offline competitions [11, 12, 13]. Besides score-level fusion, research was conducted on the combination or extension of acoustic features. Murty [14] proposed the residual phase feature as an additional feature for the MFCC feature. The bottleneck feature extraction model was introduced in [15]. The tandem feature [16] was proposed by splicing the bottleneck feature with a basic acoustic feature. In [17], two acoustic features were concatenated directly to create a new feature vector with Linear Discriminant Analysis (LDA) reducing the new feature's dimensions. In [18], an end-to-end framework was presented with an auxiliary feature learning branch.

In our previous work, we found that multiple acoustic features integration learning (MFI) that is integrated within the neural network at the frame level improved the system performance in ASV [19]. In this paper, we further investigate and compare more multiple acoustic feature integration architectures, including those (1) with two applications: ASV and LID; and those (2) with more integration levels, such as the statistics pooling level, the segment level, and the embedding level. In addition, we also expand the proposed multi-feature integration structure by introducing a time restricted-attention mechanism, namely the Attentive Multi-feature Integration (AMFI), which outperforms our former multi-feature integration architecture (MFI). The introduction of attentive learning encourages the two kinds of acoustic features to be more speaker/language discriminative and to learn more sequential information.

The rest of this paper is organized as follows. In Section 2 we describe different feature integration architectures (MFIs) and introduce the proposed multi-feature integrations with the attention mechanism (AMFIs). The experimental settings are presented in Section 3, and the experimental results are shown in Section 4. Finally, Section 5 concludes this paper.

2. Multi-feature Integration

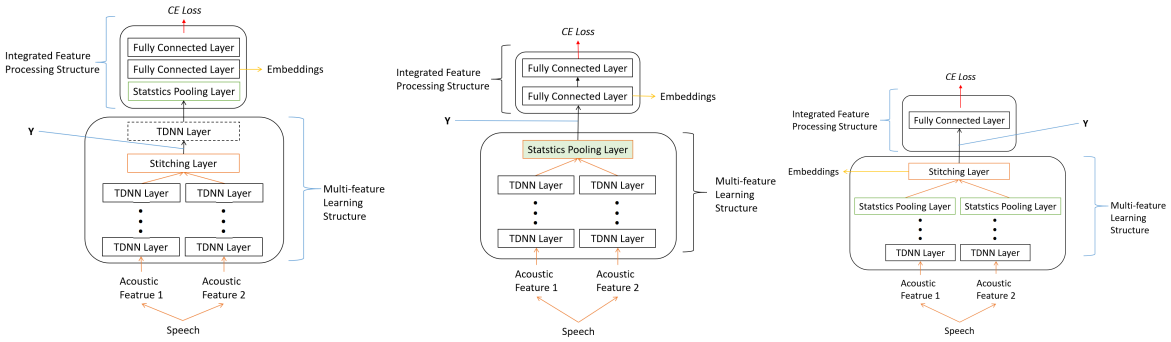
2.1. Multi-feature Integration

2.1.1. Frame Level Multi-feature Integration

To utilize the complementarity of acoustic features, we analyzed and proposed the frame-level multiple acoustic feature integration structure [19], as shown in Figure 1 (a). With this structure, two acoustic features are used to train a recognition model simultaneously, and two features are combined as one before the statistics pooling operation. Let $(X_1, X_2) \subset \mathbb{R}$ denote two kinds of acoustic feature vectors, such as vectors from MFCC and PLP, from the same speech frame, and Y_1 represents the integrated feature:

$$Y_1 = f_3\left(cat\left(f_1\left(X_1;\Theta_1\right), f_2\left(X_2;\Theta_2\right)\right);\Theta_3\right) \quad (1)$$

where $cat\left(\cdot\right)$ indicates the concatenating operation,



(a) Frame level integration. (b) Pooling level integration. (c) Segment level integration.

Figure 1: *Multiple features integration at different levels.*

$f_1(X_1;\Theta_1)$ is the pre-projection of Acoustic Feature 1 given the network parameters Θ_1, and the same for $f_2(X_2;\Theta_2)$, and $f_3(\cdot;\Theta_3)$ refers to the mapping in the fully connected stitching layer.

2.1.2. *Multi-feature Integration in Different levels*

As illustrated in 2.1.1, multiple features can be integrated at the frame level and be jointly trained within an x-vector model, which yields significant improvements in ASV tasks [19]. In this study, we further investigate the possibility of assembling feature branches at higher levels: the statistics pooling layer and the segment level, as shown in Figure 1 (b) and (c) separately. These two kinds of architectures integrate multiple features at or after the statistics pooling layer, so that each feature learns its own speaker/language discriminative information at the frame level, while being integrated at a higher level. If we let Y_2 and Y_3 represent the statistics pooling level and segment level integrated features, the computations are written as:

$$Y_2 = P_1^T\left(cat\left(f_1\left(X_{1t};\Theta_1\right), f_2\left(X_{2t};\Theta_2\right)\right)\right) \tag{2}$$

$$Y_3 = f_3\left(cat\left(P_1^T\left(f_1\left(X_{1t};\Theta_1\right)\right); P_1^T\left(f_2\left(X_{2t};\Theta_2\right)\right)\right);\Theta_3\right) \tag{3}$$

where $X_{1t} \subset \mathbb{R}$ represents the input Acoustic Feature 1 of the t^{th} frame of an utterance. The same applies X_{2t}; $f_1(X_1;\Theta_1)$ is the pre-projection of Acoustic Feature 1 given the network parameters Θ_1, and the same also goes for $f_2(X_2;\Theta_2)$. $P_1^T(\cdot)$ refers to the statistical pooling opreation that computes the mean and standard deviation accumulated from the 1^{th} to T^{th} frame, if we let the number of frames in a chunk of speech equal to T. $f_3(\cdot;\Theta_3)$ is the total computation in fully connected layers with Θ_3.

2.1.3. *Embedding Level Integration*

Feature representations can also be integrated at the embedding level, or x-vector level, after they are extracted from subsystems. For embedding level integration, we choose the most practical and easily implemented equal weight addition for embeddings, named embedding_add in Section 4. The embedding concatenation is selected as the second method to integrate embeddings, which is named embedding_cat.

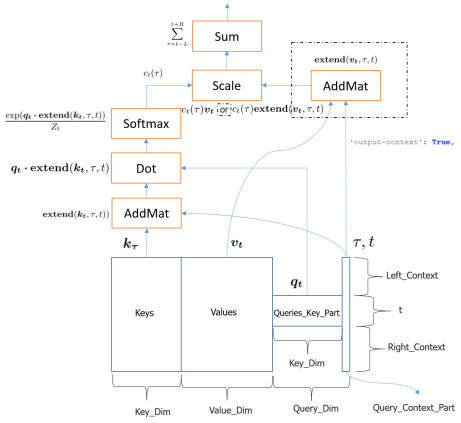
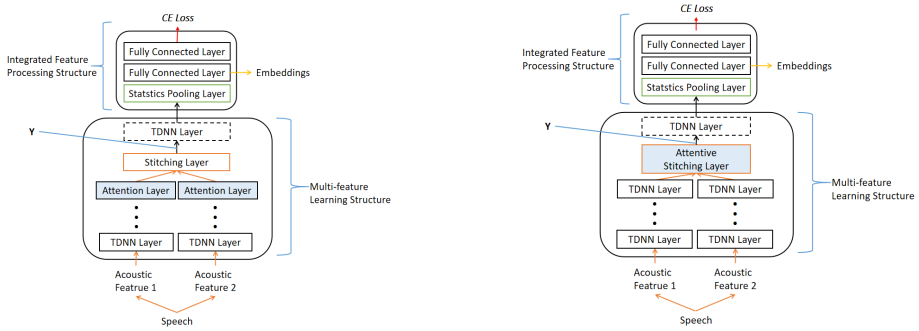


Figure 2: *The time-restricted attention mechanism.*

2.2. Multiple Feature Integration with Attention Mechanisms

After the computation of TDNN layers, feature representations (high-level features) are considered as information that is enhanced for classification. However, redundancies may still exist in the outputs between the last TDNN layers for two features. The useful information for classification may not be completely emphasized, and the sequential information is not well captured, which is critical for classification. So, we assume that introducing attention layers before the stitching layer or executing the attention with the stitching layer would be instrumental for multi-feature learning.

We design a multi-feature integration structure with a time-restricted attention mechanism, which was previously proposed for ASR, to assess multi-feature integration training [20, 21]. The time-restricted attention layer consists of an affine component, an attention nonlinearity component, and a ReLU nonlinearity component followed by batch normalization, while all the trainable parameters are in the affine component. As shown in Figure 2, in the one-head case for simplicity, the time-restricted attention utilizes related contexts, from left contexts L to right contexts R, to compute attention weights for the local frame t. The final output y_t is the accumulation of related contexts. Furthermore, the position encodings τ, t are appended in every frame with the value v_t, which is written as $extend(v_t, \tau, t)$, so that the τ, t are the one-hot encodings. To achieve better performance, we use the multi-head time-restricted attention mechanism, rather than only the one-head attention. The computation of multi-head attention is as follows:



(a) Type 1. (b) Type 2.

Figure 3: *Architectures for the multi-feature integration structure with attention mechanisms.*

$$y_t = cat\left(y_{t_1}, y_{t_2}, ..., y_{t_n}\right), n \subset \mathbb{N} \tag{4}$$

where the $cat\left(\right)$ means the concatenating operation.

The computation of the i^{th} head in the multi-head time-restricted attention mechanism is written as:

$$y_{t_i} = \sum_{\tau=t-L}^{t+R} c_t\left(\tau_i\right) \cdot extend\left(v_{t_i}, \tau_i, t_i\right) \tag{5}$$

where $extend\left(\right)$ is the concatenating operation.

The input x_t attention component is interpreted being three parts appended together: q_t, k_t, and v_t which are the query, key, and value respectively:

$$c_t\left(\tau\right) = \frac{exp\left(q_t \cdot extend\left(k_t, \tau, t\right)\right)}{Z_t} \tag{6}$$

where Z_t ensures $\sum_\tau c_t\left(\tau\right) = 1$.

We first investigate and compare the multiple level integration and the experimental results, which are reported in Section 4. These results show that the frame level integration of multiple features is the optimum strategy. Thus, for simplicity, we only introduce the attention mechanism into the frame-level integration structure.

In the case of frame level integration, the attention layer replaces the TDNN layer before the stitching layer (named type 1 attention, AMFI 1). Alternatively, we execute the attentive learning within the stitching layer (named type 2 attention, AMFI 2), as illustrated in Figure 3 (a) and (b). In type 1 attention, two high-level features have their own attention, and the above formulation (1) is rewritten as:

$$Y_{att1} = f_3\left(cat\left(att_L^R\left(f_1\left(X_1;\Theta_1\right)\right), att_L^R\left(f_2\left(X_2;\Theta_2\right)\right)\right);\Theta_3\right) \tag{7}$$

where att_L^R represents the multi-head attention learning with contexts from L to R.

In type 2 attention, the integration integrates contexts and information from two features:

$$Y_{att2} = f_3\left(att_L^R\left(cat\left(f_1\left(X_1;\Theta_1\right), f_2\left(X_2;\Theta_2\right)\right)\right);\Theta_3\right) \tag{8}$$

With the attention mechanism, multi-feature learning emphasizes the classification-related features and frames in an utterance. This is done by the computation of weights with contexts and position encodings in the attention layer and alleviating redundancies between two feature representations by enhancing the more important features. Moreover, it introduces additional context information to the higher layer's classification with the position encodings.

3. EXPERIMENTAL SETTINGS

3.1. Data

For the ASV task, the VoxCeleb 1 [22] training set and test set were utilized. Before ASV training, Kaldi's [23] augmentation recipe was used to expand the training set. 140,000 noisy utterances were randomly chosen and mixed up with the original 148,642 utterances to compose a new expanded training set. The augmented VoxCeleb 1 training set contains about 290,000 utterances with 1,211 speakers. The standard VoxCeleb 1 test set, including 4,715 utterances from 40 speakers, was used as the test set for ASV task.

For the LID task, we chose the AP-OLR17 training set [9] to build the language identification model and the AP-OLR17 task 1 (short-utterance test condition, 1 second per utterance) test set to evaluate the countermeasures. No extra data were used to augment the training data for the LID task, but we used speech perturbing to extend the LID training set 3-fold (0.9, original, and 1.1). The augmented AP-OLR17 training set has about 280,000 utterances with ten original languages, including Mandarin, Cantonese, Indonesian, Japanese, Russian, Korean, Vietnamese, Kazakh, Tibetan and Uyghur. The LID test set included 22,051 utterances from ten target languages.

3.2. System Settings

In our experiments, two combinations of the typical, popular acoustic features, MFCC-FBank and MFCC-PLP, were chosen to investigate the performance of the proposed multi-feature integrations in ASV and LID tasks.

All models were trained on 16kHz audio data. The acoustic features included 30 dimensional MFCC features, 30 dimensional PLP features and 40 dimensional FBank features in all systems. In LID systems, additional three-dimensional pitch features were appended to acoustic features. All features had frame-lengths of 25ms, frame-shifts of 10ms, and mean normalizations over a sliding window of up to three seconds. Voice active detection (VAD) was used to filter out non-speech frames, and note that different acoustic features share the same VAD for the frame alignment.

The network architecture of the x-vector baseline system was the same as in [6], and all models were optimized based on the x-vector architecture. Consequently, we extracted embeddings from the penultimate layer after training. For the time-

Table 1: *Experimental results with the metric value EER(%).*

System		ASV Task		LID Task	
Baseline (PLP)		4.67		10.04	
Baseline (FBank)		5.35		10.34	
Baseline (MFCC)		4.76		10.76	
No.	Feature Combination	MFCC PLP	MFCC FBank	MFCC PLP	MFCC FBank
1	MFI @ 1st	3.95	3.83	9.64	9.20
2	MFI @ 2nd	4.08	3.62	8.88	9.48
3	MFI @ 3rd	3.76	3.59	8.68	9.04
4	MFI @ 4th	3.86	3.70	8.57	9.04
5	MFI @ 5th	**3.73**	**3.48**	**8.47**	**8.97**
6	MFI @ stats	3.73	3.69	8.83	9.35
7	MFI @ 6th	3.81	3.60	8.84	9.12
8	Embedding_add	4.37	4.68	9.05	9.38
9	Embedding_cat	4.23	4.56	9.13	9.42
10	Score Level	4.23	4.19	9.06	8.95
11	AMFI 1	3.63	3.48	8.51	8.95
12	AMFI 2	**3.53**	**3.38**	**8.12**	**8.48**

restricted attention layer, we used 20-headed attention with the extension operation. The contexts L and R were set to three, and the dimensions of the key k_t and value v_t were 40 and 60, respectively. All models were trained on Kaldi with the SGD optimizer.

For ASV tasks, the back-end process was the same as Kaldi's Voxceleb 1 recipe, which included the LDA, centering and PLDA. In the LID task, we chose logistic regression (LR) as the classifier after carrying out the back-end process, which included the LDA, whitening, and centering. Back-ends were implemented on the Kaldi platform.

4. RESULTS AND ANALYSIS

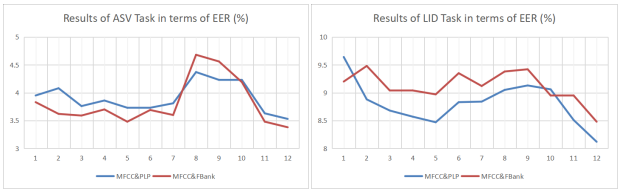
Experimental results are listed in Table 1 in terms of EER(%).

4.1. Speaker Recognition Task

From the comparison of embedding-level fusion and score-level fusion in baselines, the combination of MFCC and FBank yielded greater improvements in the ASV task than the combination of MFCC and PLP. Feature integration within the neural network achieved significant improvements compared with the baseline systems in both kinds of feature combinations. The best performing systems were frame level integrations, while the segment level integration and the integration in the statistics pooling layer offered much less improvement. When the model with the stitching layer at the frame level integration was chosen appropriately, those systems outperformed systems at other integration levels. For the integration pattern of MFCC and FBank, when branches were stitched at the fifth layer, the best EER was 3.48%, which was 26% relative improvement over the best baseline. On the other hand, the embedding level fusion could not reliably promise much improvement. In two kinds of implementations of attentive integrations, AMFI 1 and AMFI 2, the attention layer contributed to the multi-feature integration ASV system, while AMFI 2 achieved greater improvement. The best EER among the system with the attention mechanism was 3.38%, which was 28% relative improvement over the best baseline system.

4.2. Language Identification Task

The LID systems with the integration pattern of MFCC and FBank outperformed systems with the integration pattern of MFCC and PLP; these were shown at the baseline embedding level and score level integration. Similar to the results in the speaker recognition task, the best performing system was also achieved at the frame level. The best stitching layer was also the fifth layer with 8.47% EER, which obtained 16% relative improvement over the best baseline. This performance surpassed the score level fusion on baselines. Likewise, the embedding level fusion strategy was not the most effective method. The proposed AMFIs definitely improved the performance of the LID task. The improvements outperformed the other integration systems, while AMFI 2 was better than AMFI 1 in the LID task, reaching 8.12% in terms of the EER value.



(a) Results of ASV Task. (b) Results of LID Task.

Figure 4: *Trends of mutli-feature integration systems' experimental results.*

4.3. Similarities in Two Speech Classification Tasks

The trends in the results are shown in Figure 4, in which the horizontal ordinates indicate the number of systems corresponding to Table 1, and the Y-ordinates show the values of EER (%). From the comparisons of Table 1 and Figure 4, in spite of the region of EER values that are different between the two tasks, there were some similarities in the two classification tasks. For MFI integration patterns, the best configuration was the frame level integration on the 5^{th} layer. While, the higher level integrations cannot achieve better performance than the frame level integration in MFIs. Moeover, the AMFIs yielded the best performances compared with MFIs and baselines, and AMFI 2 structure obtained superior performance.

5. Conclusions

In this study of ASV and LID tasks, we investigate the performances of multi-level integrations for acoustic features and employed the time-restricted attention mechanism in the multi-feature integration structure. In our experiments, we found that integrating multiple acoustic features at the frame level, especially the 5^{th} layer, contributes the most in both tasks. This performance surpassed score level fusion and embedding level fusion. The proposed attentive multi-feature integration architecture achieved 28% and 19% relative improvement over the best baselines in ASV and LID, respectively. The trends in the experimental results for ASV and LID were similar for feature integration, which indicates that the multi-feature integration strategy can be generalized for those two speech classification tasks. In the future, we plan to focus on finding more potential multi-feature learning strategies.

6. Acknowledgements

This work is supported by the National Natural Science Foundation of China (Grant No.61876160).

7. References

[1] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted Gaussian mixture models," *Digital signal processing*, vol. 10, no. 1-3, pp. 19–41, 2000.

[2] P. Kenny, "Joint factor analysis of speaker and session variability: Theory and algorithms," *CRIM, Montreal,(Report) CRIM-06/08-13*, vol. 14, pp. 28–29, 2005.

[3] 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.

[4] Y. Lei, N. Scheffer, L. Ferrer, and M. McLaren, "A novel scheme for speaker recognition using a phonetically-aware deep neural network," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 1695–1699.

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

[6] 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.

[7] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE transactions on acoustics, speech, and signal processing*, vol. 28, no. 4, pp. 357–366, 1980.

[8] H. Hermansky, "Perceptual linear predictive (PLP) analysis of speech," *the Journal of the Acoustical Society of America*, vol. 87, no. 4, pp. 1738–1752, 1990.

[9] Z. Tang, D. Wang, and Q. Chen, "AP18-OLR challenge: Three tasks and their baselines," in *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2018, pp. 596–600.

[10] M. Zhao, R. Li, S. Yan, Z. Li, H. Lu, L. Li, and Q. Hong, "Phone-aware multi-task learning and length expanding for short-duration language recognition," in *APSIPA 2019*. IEEE.

[11] L. J. Rodríguez-Fuentes, N. Brümmer, M. Penagarikano, A. Varona, M. Diez, and G. Bordel, "The Albayzin 2012 language recognition evaluation plan (Albayzin 2012 LRE)," *URL: http://iberspeech2012. ii. uam. es/images/PDFs/albayzin_lre12_evalplan_v1. 3_springer. pdf*, 2012.

[12] N. Brümmer and E. De Villiers, "The Bosaris Toolkit: Theory, algorithms and code for surviving the new DCF," *arXiv preprint arXiv:1304.2865*, 2013.

[13] N. Brümmer, "Focal multi-class: Toolkit for evaluation, fusion and calibration of multi-class recognition scorestutorial and user manual," *Software available at http://sites. google. com/site/nikobrummer/focalmulticlass*, vol. 33, p. 39, 2007.

[14] K. S. R. Murty and B. Yegnanarayana, "Combining evidence from residual phase and mfcc features for speaker recognition," *IEEE signal processing letters*, vol. 13, no. 1, pp. 52–55, 2005.

[15] S. Yaman, J. Pelecanos, and R. Sarikaya, "Bottleneck features for speaker recognition," in *Odyssey 2012-The Speaker and Language Recognition Workshop*, 2012.

[16] M. McLaren, Y. Lei, and L. Ferrer, "Advances in deep neural network approaches to speaker recognition," in *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2015, pp. 4814–4818.

[17] Z.-Y. Li, L. He, W.-Q. Zhang, and J. Liu, "Multi-feature combination for speaker recognition," in *2010 7th International Symposium on Chinese Spoken Language Processing*. IEEE, 2010, pp. 318–321.

[18] F. Zhao, H. Li, and X. Zhang, "A robust text-independent speaker verification method based on speech separation and deep speaker," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6101–6105.

[19] Z. Li, H. Lu, J. Zhou, L. Li, and Q. Hong, "Speaker embedding extraction with multi-feature integration structure," in *APSIPA 2019*. IEEE.

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[21] D. Povey, H. Hadian, P. Ghahremani, K. Li, and S. Khudanpur, "A time-restricted self-attention layer for ASR," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5874–5878.

[22] A. Nagrani, J. S. Chung, and A. Zisserman, "VoxCeleb: a large-scale speaker identification dataset," *arXiv preprint arXiv:1706.08612*, 2017.

[23] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The Kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*. IEEE Signal Processing Society, 2011.