The UJS System for The Far-Field Speaker Verification Challenge 2020

You-Cai Qin[1], *Qi-rong Mao*[1,2]

[1] Computer Science and Communication Engineering, Zhenjaing, Jiangsu Province, 212000, China
[2] Jiangsu Key Laboratory of Security Tech. for Industrail Cyberspace
221908026@stmail.ujs.edu.cn, mao_qr@ujs.edu.cn

Abstract

This paper describes the systems developed by the UJS team for the Far-Field Speaker Verification Challenge 2020 (FFSVC). We participate in task 1 of this challenge. The performance of speaker verification is degraded by the presence of large amounts of reverberation and uncertain background noise in far-field environments. In order to solve this problem, we have adopted many methods, which include data augmentation, adversarial multi-task training, transfer learning and model fusion. The best single model uses ResNet34, along with AAM-SoftMax. Finally, using the method of model fusion, we achieve 0.57 minDCF in the evaluation set and the performance is better than baseline.

Index Terms: Speaker Verification, Far-field, deep learning

1. Introduction

Speaker verification in a certain environment has achieved satisfactory performance. However, in some complex scenarios, such as unknown noise and far-field environment, the performance of speaker verification is degraded. In the far-field environment, sound signal is reflected by obstacles to form reverberation, which reduces the quality of the speech signal. The background of the real environment contains a variety of noises, such as TVs, other speakers, etc. Speaker verification in complex environments is still challenging. Recently, speaker recognition databases containing complex environments [1,2] have been released to help researchers better study this challenge.

In early work, the most dominant method of speaker validation is i-vector/PLDA [3], widely used in both industry and academia. However, this model is linear, shallow, and cannot be applied in complex environments. Recently, deep speaker embedding networks obtains better performance and gradually replacing i-vector/PLDA.

X-vector [4] is a representative model of deep speaker embedding. It extracts speech frame-level features through TDNN and uses statistical pooling functions to generate utterance-level representations. Convolutional neural-based networks have also achieved good performance in speaker verification tasks [5]. In addition, there are LSTM-based, CRNN-based models that are also used for speaker validation.

In order to improve the performance of speaker verification in complex environments, several approaches have been proposed for noise reduction and de-reverberation. Weighted prediction error (WPE) is an effective way to de-reverberate [6]. In [7], speech separation is used to reduce noise and improve the performance of speaker verification. There are also approaches that use adversarial training to generate noise-invariant, channel-invariant representations [8,9].

The Far-Field Speaker Verification Challenge 2020 (FFSVC20) is designed to boost the speaker verification research with special focus on far-field distributed microphone arrays under noisy conditions in real scenarios [10]. Our system pipeline consists of five main components, including front-end processing, deep speaker embedding network, back-end processing, Transfer learning (text-independent speaker verification model) and adversarial multi-task training.

This paper is organized as follows: Section 2 describes the details of our submitted system. Section 3 clarifies the experimental results and analysis. Conclusions are drawn in section 4.

2. System descriptions

Our system consists of the following five main components, including front-end processing, deep speaker embedding network, back-end processing, transfer learning (text-independent speaker verification model) and adversarial multi-task training. The description of each components is as follows.

2.1. Front-end processing

2.1.1. Voice activity detection

VADs are often used at the front end of various speech tasks because they remove the silence, reduce the amount of computation and improve the voice quality. In our system, all speech is removed from silent segments via VAD.
We adopted the energy-based VAD method. The principle of energy-based VAD is that the vocal part of the speech spectrum tends to have a higher energy than the silent part of the background.

2.1.2. Acoustic features

The characteristics of speech signals are often reflected in the frequency domain, speaker verification and speech recognition often use time-frequency domain characteristics, such as Mel-frequency cepstral coefficient (MFCC), Log Mel-filterbank energies (Fbank).
In our system, Fbank is used as an input to the model. All features are extracted from 25ms windows with 10ms shift between frames. Each audio is converted to 64-dimensional log Mel-filterbank energies with cepstral filterbanks ranging from 0 to 16000 Hz. To be able to speed up the training process, mean normalization is used to normalize the data.

2.1.3. Data augmentation

Although there is a large amount of training data, most of the data is the same piece of speech being recorded by different devices. Therefore, these utterances are very similar, and it is easy to overfit the training model. Multiple approaches to data augmentation need to be used. The data augmentation methods include mixed noise, far-field environment simulation, multi-scale training.

Mixing training data and noise according to a random signal-to-noise ratio can improve sample diversity. We used three different types of noise (music, speech, and noise) from the open source noise library [11]. The signal-to-noise ratio is selected in the range of 5-20 db.

Closed-field speech is simulated as far-field speech to augment data via pyroomacoustics [12]. The addition of far-field data can help the network better understand the far-filed data and learn about robust embedding. The parameters of the simulation are set to the height of the room to be 3 meters, and randomly select 3-12 meters as the length and width of the room. Speakers and microphones are randomly placed in the room.

Different lengths of data are used as input during the training process. In each training process, the number of frames of speech randomly ranges from 80-160 frames. Different sized datasets help the model capture important passages in speech, while expanding the dataset improves the overfitting problem to some extent.

2.2. Deep speaker embedding network

As shown in Figure 1, the deep speaker model mainly consists of four components, including feature extraction, pooling module, fully connected layer and loss function.

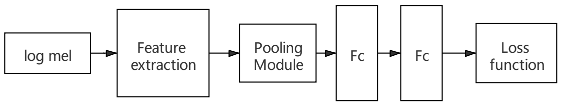

Figure 1: system architecture

2.2.1. *Feature extraction*

The feature extractor converts the speech sequences into frame-level representations. We extract the feature extraction from two state-of-the-art modelings, the deep ResNet system and the TDNN x-vector system.

The similar deep ResNet structure in the baseline is used as a feature extractor [13]. The number of channels of the residual block is {32-64-128-256}. The detailed configuration of the deep ResNet architecture is in Table 1. Some details about TDNN networks can be found in [7]. The difference between TDNN x-vector and ResNet is that the former uses TDNN and the latter uses convolutional networks. The implementation of the two architectures both use Pytorch.

2.2.2. *Pooling module*

The function of the pooling module is to convert frame-level speech representations into utterance-level representations. Recently frequently used pooling methods include Global average pooling (GAP), Global statistics pooling (GSP), Temporal attention pooling.

The pooling module in our system uses GSP. The GSP calculates the mean and variance in each channel and connects them into vectors. More dimensions of variance than GAP, with better robustness in complex environments. Although the length of the utterance is not the same at the time of input, after the pooling function, all the utterance is converted to the same size representation.

Table 1: The deep ResNet and Pooling module

Layer	**Output Size**	**Structure**
Conv1	32*64*L	$C(3*3,1)$
Residual Layer 1	32*64*L	$\begin{bmatrix} C(3*3,1) \\ C(3*3,1) \end{bmatrix} * 3$
Residual Layer 2	64*32*$\frac{L}{2}$	$\begin{bmatrix} C(3*3,2) \\ C(3*3,1) \\ S(1*1,2) \end{bmatrix} \begin{bmatrix} C(3*3,1) \\ C(3*3,1) \end{bmatrix} * 3$
Residual Layer 3	128*16*$\frac{L}{4}$	$\begin{bmatrix} C(3*3,2) \\ C(3*3,1) \\ S(1*1,2) \end{bmatrix} \begin{bmatrix} C(3*3,1) \\ C(3*3,1) \end{bmatrix} * 5$
Residual Layer 4	256*8*$\frac{L}{8}$	$\begin{bmatrix} C(3*3,2) \\ C(3*3,1) \\ S(1*1,2) \end{bmatrix} \begin{bmatrix} C(3*3,1) \\ C(3*3,1) \end{bmatrix} * 2$
Pooling module	512	Global Statistics Pooling
Embedding	128	Fully connected Layer
Classifier	460	Fully connected Layer

2.2.3. *Loss function*

In speaker verification, a combination of SoftMax and cross-entropy is often used as a loss function. SoftMax encourages separation of features between classes, but does not require intra-class compactness or inter-class separation. For this reason, SoftMax may not have learned the highly discriminating embedding

Recently, AAM-SoftMax, improved from SoftMax, has been proposed to obtain very good performance in face verification. In angular space, AAM-SoftMax maximizes classification boundaries and can effectively reduce the intra-class distances and increase inter-class distances. AAM-SoftMax is defined as:

$$L_{AAM-softmax} = -\frac{1}{N}\sum_{i=1}^{N} log \frac{e^{s\left(cos\left(\theta_{y_i,i}+m\right)\right)}}{Z} \tag{1}$$

$$Z = e^{s\left(cos\left(\theta_{y_i,i}\right)+m\right)} + \sum_{j=1,j\neq i}^{C} e^{s\left(cos\left(\theta_{j,i}\right)\right)} \tag{2}$$

SoftMax and AAM-SoftMax are used in our system. We compared their performance for the far-field speaker verification task.

2.3. Back-end processing

2.3.1. *cosine similarity*

Each utterance is extracted as a 128-dimensional embedding. We use cosine similarity to determine the similarity between two embedding.

Suppose the embedding of the registered speech is E_{reg}, and the embedding of the test speech is E_{test}. The cosine similarity is defined as:

$$\text{Score}_{(\text{reg,test})} = \cos(\theta) = \frac{E_{reg}\cdot E_{test}}{\|E_{reg}\|\|E_{test}\|} \tag{3}$$

2.3.2. *model fusion*

We utilize model fusion at the score level. A linear model is used to weight the scores of different models to obtain the final score. The experimental results show that the model fusion approach is effective in improving performance.

2.4. Transfer learning

According to the baseline [13], Fine-tuning with a pre-trained text-independent speaker recognition model yields better results. We pre-trained the deep speaker network with larger scale text-independent mix-dataset (close-talk and its simulation data). The pre-training data contained 2015 speakers, including five Chinese databases from openslr.org. The five databases are SLR33, SLR38, SLR47, SLR62, and SLR68. The fine-tuning data is SLR85 dataset and the first 30 utterances of FFSVC 2020 training dataset.

2.5. Adversarial multi-task training

In the challenge task, the enrolled utterance is recorded using a close cellphone and the test utterance is recorded using a microphone array at a distance of 1 to 5 meters. There is a device mismatch between the enrollment and test sets. Meanwhile, there is variability between the far-field microphone arrays recording speech from different locations.

We use an adversarial multitasking approach to address enrollment and test speech mismatches and far-field microphone locations changes. A classifier for discriminating devices and locations is added to the original network for adversarial multitasking training. Minimize the loss of speaker verification while maximizing the loss of the classifier for devices and locations during training. The goal of adversarial multi-task training is to make it impossible for the network to determine exactly which device the speech belongs to and at which location the microphone array is recorded.

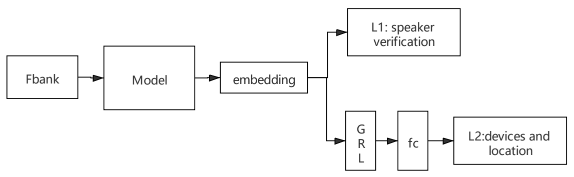

Figure 2: Adversarial multi-task architecture

As shown in Figure 2, Adversarial multitasking module requires a gradient reversal layer (GRL), a fully connected layer and a classifier. GRL lets the gradient computed by the classifier reverse back to the model, creating an adversarial situation between the classifier for devices and locations and the embedding. The proposed model is able to learn device-invariant and position-invariant speaker representations via adversarial multitasking training.

Devices and locations are jointly coded to 5 categories: close-filed categories (High quality microphone, cellphone), far-filed categories (1m microphone arrays, 3m microphone arrays, 5m microphone arrays).

Assuming that the loss function for speaker verification is L_1 and the loss function for classifier for devices and locations is L_2, then the final loss function is defined as:

$$L = L_1 + \alpha * L_2 \tag{4}$$

where α is a hyper-parameter and controls the trade-off between the speaker verification loss and the classifier for devices and locations loss.

3. Experiment

3.1. Text-independent speaker verification training

We train the text-independent speaker verification system using five open-sourced databases from openslr.org, which are the data allowed for this challenge. Speakers in these databases who have fewer than 80 utterances are deleted, and the final data available for training contain a total of 2015 speakers. Because the training set is cobbled together from multiple datasets, the number of utterances available for training per speaker is unbalanced. We sample the data balance for each speaker during training. In each training epoch, each speaker selects 10 randomly utterance for training.
Frame-level representations are extracted using ResNet or TDNN, as described previously. GSP is used as pooling module. The loss function uses a simple SoftMax. In a training epoch, the number of utterances is 20,150, And each epoch takes approximate 100s to train on a NVIDIA TESLA P100 16GB device. The model was trained for 1000 epochs. The initial learning rate is 1e-3, decreasing to 1e-4 at 500 epochs. The batch-size set to 64. For each epoch, the number of frames of training data is randomly selected from 100-200.

3.2. Text-dependent speaker verification training(fine-tune)

The trained text-independent speaker model is used as a pre-trained model. Its parameters are used as initialization parameters for text-dependent speaker verification. A pre-training model is a good starting point for training and speeds up the training process.
The first 30 utterances of each speaker in the official train dataset and the SLR85 database from openslr.org are used to fine-tune. The content of all utterance is "hi, mia" in Chinese or English. The database used for fine-tuning contained a total of 460 speakers, with each speaker from SLR85 containing approximately 3900 utterances and each speaker from the official training set containing approximately 1200 utterances. The former provides 16 channels of microphone array data, while the latter only has 4 channels of microphone array data. Therefore, dataset for training is not balanced. As with text-independent speaker identification, the data is resampled. In each training epoch, each speaker randomly selects 10 utterances for training.
Text- dependent speech is usually shorter than text-independent speech, thus reducing the number of frames relative to text-independent speaker training. In each epoch, speech was randomly truncated or spliced into 100-140 frames. Text-dependent speaker verification model was trained for a total of 600 epochs. The batch-size also set to 64. The two parameters contained in the AAM-SoftMax are set to s = 0.2, m = 30. The initial learning rate is set to 1e-3, decreasing to 1e-4 at 200 epochs and 1e-5 at 400 epochs. The model takes up 19.3M of storage space.
In adversarial multitasking training, the classifier for devices and locations are trained, SoftMax and cross-entropy are used as loss functions. When the hyperparameter α in the final loss function is set to 0.001, the model reaches the best performance.

Table 2: Performance of the speaker verification systems. "Model" represents the types of deep speaker embedding network; "TL" represents whether to use pre-trained models for transfer learning; "AMT" represents whether to use adversarial multi-task training.

ID	Model	TL	Loss function	AMT	Task1 Development set		Task1 Evaluation Set	
					minDCF	EER	minDCF	EER
1	Baseline(resnet34)	Y	SoftMax	N	0.57	6.01%	0.62	6.37%
2	ResNet34	N	AAM-SoftMax	N	0.66	7.31%	-	-
3	ResNet34	Y	AAM-SoftMax	N	0.53	5.55%	0.61	6.41%
4	ResNet34	Y	AAM-SoftMax	Y	0.51	5.37%	0.59	6.47%
5	TDNN	N	SoftMax	N	0.73	8.65%	-	-
6	TDNN	N	AAM-SoftMax	N	0.76	9.89%	-	-
7	Fusion1(2+3+4+5)	-	-	-	0.46	**4.73%**	0.57	5.83%
8	Fusion2(2+3+4+5)	-	-	-	**0.44**	4.87%	**0.57**	**5.78%**

3.3. result and analysis

The experimental results on the far-field speaker verification system are shown in Table 3. ID1 is the baseline model provided by [13] and ID2-8 is the model submitted by our team system. Feature extraction includes Resnet34 and TDNN, while a number of other configurations are also explored. For example, whether to use a pre-trained model for transfer learning, what loss function to use, whether to use adversarial multi-task training.

As can be seen from the table, the TDNN-based approach is not as effective as the Resnet-based approach. This suggests that TDNN is less capable of extracting representations than ResNet in our speaker validation system.

Based on the results of ID2 and ID3, it can be found that transfer learning using pre-trained models is very effective. Knowledge learned in the pre-training model can help text-dependent speakers learn more discriminative information. Text-dependent speaker verification uses the parameters of a pre-trained model as initialization parameters, which can speed up training and prevent model overfitting.

Of all the simple models, the models with the best results is ID5. ID5 adds the adversarial multitasking training compared to model ID4. In the development set, ID5 improved minDCF by 3.7% compared to ID4. In the evaluation set, ID5 is also better than ID4, improving by 3.3% in minDCF. It turns out that the model is made to learn to device-invariant and location-invariant embedding using classifiers for devices and locations as well as GRL. Adversarial multitasking training is able to solve the devices mismatch problem to some extent.

ID7 performs model fusion on the scoring results of models ID2, ID3, ID4, ID5. ID8 also performs fusion on the scoring results of models ID2, ID3, ID4, ID5. The difference between ID7 and ID8 is in the inputs of the scoring, the former contains 4 scores as inputs and the latter contains 16 scores as inputs. The reason for the different input sizes is that the test utterance contains four channels. In ID7, each sub-model averages the data from the four channels and computes one score, and in ID8, each sub-model computes four scores from the four channels. As can be seen from the table, ID8 is better than ID7 on the minDCF of the development set, however, it is lower than ID7 on the EER metric. In the evaluation set, the effect of ID8 is slightly better than that of ID7 on both metrics, but the difference is not great. Experimental results show that model fusion can effectively improve the performance of the system.

Compared with the baseline, ID8 has increased by 8.7% minDCF and 10.2% EER in the evaluation set.

4. Conclusions

This paper describes the UJS team's system for Far-Field Speaker Verification Challenge 2020. ResNet and TDNN are used as feature extractors. The results show that the ResNet-based feature extractor works better than TDNN in our system. We still use two different loss functions, SoftMax and AAM-SoftMax. For enrolling and testing speech from different devices and different locations between microphone arrays, adversarial multitasking training is used and a classifier of devices and locations is proposed. By maximizing the error of this classifier, the model is capable of generating device-invariant and location-invariant representations. The best results come from scoring model fusion. This shows that model fusion is a great way to improve performance. In the future, we will further investigate the robustness of speaker verification in complex environments

5. References

[1] Qin X, Bu H, Li M. Hi-mia: A far-field text-dependent speaker verification database and the baselines. ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2020: 7609-7613.

[2] Fan Y, Kang J W, Li L T, et al. CN-CELEB: a challenging Chinese speaker recognition dataset. ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2020: 7604-7608.

[3] Viñals I, Gimeno P, Ortega A, et al. Estimation of the Number of Speakers with Variational Bayesian PLDA in the DIHARD Diarization Challenge. Interspeech. 2018: 2803-28071

[4] Snyder D, Garcia-Romero D, Sell G, et al. X-vectors: Robust dnn embeddings for speaker recognition. 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2018: 5329-5333.

[5] Qin X, Cai D, Li M. Far-Field End-to-End Text-Dependent Speaker Verification Based on Mixed Training Data with Transfer Learning and Enrollment Data Augmentation. INTERSPEECH. 2019: 4045-4049.

[6] Nakatani T, Yoshioka T, Kinoshita K, et al. Speech dereverberation based on variance-normalized delayed linear prediction. IEEE Transactions on Audio, Speech, and Language Processing, 2010, 18(7): 1717-1731.

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

[8] Meng Z, Zhao Y, Li J, et al. Adversarial speaker verification. ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2019: 6216-6220.

[9] Fang X, Zou L, Li J, et al. Channel adversarial training for cross-channel text-independent speaker recognition. ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2019: 6221-6225.

[10] Qin X, Li M, Bu H, et al. The FFSVC 2020 Evaluation Plan. arXiv preprint arXiv:2002.00387, 2020.

[11] Snyder D, Chen G, Povey D. Musan: A music, speech, and noise corpus. arXiv preprint arXiv:1510.08484, 2015.

[12] Scheibler R, Bezzam E, Dokmanić I. Pyroomacoustics: A python package for audio room simulation and array processing algorithms. 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2018: 351-355.

[13] Qin X, Li M, Bu H, et al. The INTERSPEECH 2020 Far-Field Speaker Verification Challenge. arXiv preprint arXiv:2005.08046, 2020.