Speaker recognition based on D-vector and language recognition

Chuyu Zhang ,Mingtao Wang
Xiaobo Li ,Yueyang Hu

Abstract— **The effects of different features and classifiers on language recognition and speaker recognition are investigated in the experiment. The method of using GMM-UBM to classify speech features based on PLP and MFCC is applied to language recognition, and the accuracy rate can reach more than 96%. The d-vector feature extraction methods based on DNN, GRU and LSTM can achieve 43.7% accuracy in speaker recognition. D-vector model based on speaker recognition is directly applied to language recognition with 98.1% accuracy, which proves that d-vector has strong generalization ability. At the same time, based on the difference of GRU and LSTM structure, we add a convolution layer to lstm to explore the effect of convolution layer.We found convolution layer improve the accuracy to 52.2%. GUI interface is designed and manufactured to improve interaction.**

Keywords: GMM-UBM, PLP, d-vector, MFCC

I. INTRODUCTION

Speaker and language recognition algorithms seek to determine the identity of a speaker and a language from audio. Speaker and language recognition task are classifying the identity of an unknown voice among a set of speakers or language. Verification algorithms may require the speaker to utter a specific phrase (text-dependent recognition) or be agnostic to the audio transcript (text-independent recognition). In all these tasks, use the feature extraction method to map utterances into a feature space where distances correspond to speaker or language similarity. Though many algorithms have pushed the state-of-the-art over the past couple years, speaker and language recognition is still a challenging task. The traditional recognition approach entails using MFCC or PLP. At present, the new method is to use neural network to extract speech features and identify them. Whatever the traditional method or the neural network method, the framework can be decomposed into three stages: Collect voice statistics; Extract speech features; Classify.

In this work, we use the traditional methods to achieve language recognition and neural network method to achieve speaker recognition. In language recognition, voice features use MFCC and PLP. After obtaining the PLP characteristic parameters, in order to enable the computer to recognize different speakers, we need to extract the target user's sound extraction features into one or more models and store them in the model library. The model we select is GMM-UBM (Gaussian Mixture Model - General Background Model). Because GMM-UBM gives a good pre-estimation of the probability model of spatial distribution of speech features

In speaker recognition, first we extract the speech MFCC features then use these features to extract D-vector with DNN, LSTM and GRU. These neural networks have the powerful feature extraction capability and has been successfully applied to speech recognition. At last, we compute the cosine distance between the test D-vector and the claimed D-vector and then an identification decision is made by comparing the distance to a threshold.

II. LANGUAGE RECOGNITION

The principles and methods of language recognition are as follows:

A. VAD

Voice activity detection(VAD) is also called voice endpoint detection and voice boundary detection. Its purpose is to recognize and eliminate mute period from speech signal. For a recorded voice, not the whole audio file contains all the information we need. The gap or mute period of the voice does not contain any information we need. Such silence period will not only waste our data operation resources and increase the operation time, but also reduce the recognition accuracy for speech recognition. Endpoint detection is to determine the starting and ending points of speech accurately in a signal containing voice, and to separate the voice segment from the non-voice segment.

In order to reduce the influence of noise in audio and improve the accuracy of speaker recognition and language recognition with noise frequency, we first detect the voice endpoints of the audio files used in the experiment, then cut out the part identified as silent, and stitch the remaining parts together to generate the audio files to be tested.

Two thresholds are used to detect speech endpoints. The two thresholds are short-term energy of speech and short-term zero-crossing rate. The reason for choosing these two characteristics is that the energy of voiced voice is higher than that of voiced voice, and the zero-crossing rate of voiced voice is higher than that of silent voice. So we can use the threshold of energy to distinguish the voiced part, and then use the zero-crossing rate to extract the voiceless part. This completes the endpoint detection function.

Considering that the speech signal is quasi-steady-state, the speech segment in the 20-30ms time range can be approximately regarded as steady-state signal, so the input speech signal is subdivided into frames according to 20ms per frame. The short-time energy and short-time zero-crossing rate of each frame are calculated. By multiplying each two adjacent sampling values in a frame, the number of zero-crossing points in a frame is recorded.

Comparing the short-time energy and short-time zero-crossing rate with the pre-set threshold value, the frame below the threshold value is cut off from the speech segment, and the part above the threshold value is retained. It can complete the VAD processing of the audio to be tested.

B. MFCC

The human ear has different perceptual abilities to speech at different frequencies. It is found that the perception ability is linear with frequency below $1000Hz$, and the perception ability is logarithmically related to frequency above $1000Hz$. In order to simulate the perception characteristics of the human ear to different frequencies, the concept of Mel frequency is proposed. The meaning is: 1Mel is $1/1000$ of the degree of pitch perception of $1000Hz$, and the conversion formula between frequency f and Mel frequency B is:

$$Mel(f) = 2595 \lg(1 + f_{Hz}/700) \tag{1}$$

Where f is the frequency and the unit is Hz.

The Mel Frequency Cepstral Coefficient (MFCC) is proposed based on the above-mentioned Mel frequency concept[1]. The proposed process and calculation process are shown in Fig.1 .

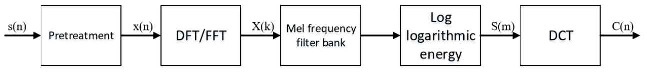

Fig. 1. the proposed process of MFCC

1) : The original signal s(n) is pre-emphasized, framed, windowed to the time domain signal x(n) of each speech frame.

2) : After the time domain signal x(n) is complemented by a number of 0, a sequence of length N (N should be an integer power of 2, where N is 256) is formed, and then a linear spectrum X (k) is obtained by Discrete Fourier Transform[6], the conversion formula is:

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi nk/N} (0 \leq n, k \leq N-1) \tag{2}$$

In practical applications, it is often calculated by a Fast Fourier Transfor, where N is generally referred to as the window length of the FFT.

3) : The linear spectrum X(k) is obtained by a Mel frequency filter bank to obtain a Mel frequency, and a logarithmic spectrum S(m) is obtained by processing a logarithmic energy, where in the Mel frequency filter bank is set in a frequency spectrum range of the speech. A number of bandpass filters $H_m(k)$, $0 \leq m \leq M$, M is the number of filters, each filter has a triangular filter characteristic, and its center frequency is $f(m)$, when the m value is small The spacing between $f(m)$ is also small. As m increases, the interval between adjacent $f(m)$ also increases.

4) : Calculate the logarithmic energy of each filter bank output:

$$S(m) = \ln\left(\sum_{k=0}^{N-1} |X(k)|^2 H_m(k)\right), (0 \leq m < M) \tag{3}$$

5) : The above-mentioned logarithmic spectrum S(m) is subjected to discrete cosine transform DCT to the cepstrum domain to obtain the Mel frequency cepstral coefficient C(n):

$$c(n) = \sum_{m=0}^{M-1} S(m) \cos\left(\frac{\pi n(m+1/2)}{M}\right), (0 \leq m < M) \tag{4}$$

In the actual application process, the number of Mel filter banks is taken as 24, and the first 13 coefficients are selected.

C. PLP

Perceptual linear prediction (PLP) is a feature parameter based on auditory model. PLP is similar to LPC[3] in that it is obtained by predictive coefficients, but PLP is a set of coefficients that can be used to predict polynomials by using all-pole model. Different from LPC, PLP technology applies some conclusions from human ear auditory test to spectrum analysis, and replaces the time-domain signal in linear prediction analysis with the signal obtained from the analog human ear auditory model.

PLP technology mainly imitates the auditory perception mechanism of human ear at three levels:

(a) Critical-band spectral resolution

(b) The equal-loudness pre-emphasis

(c) The intensity-loudness power law of hearing

1) PLP feature extraction flow chart:

Fig.2 is the principle block diagram of PLP feature parameter extraction. It can be seen that all kinds of characteristics of human ear hearing are processed by engineering and simulated by simple model. The spectrum obtained after processing is more in line with the characteristics of human ear hearing, so that the feature parameters have better robustness.

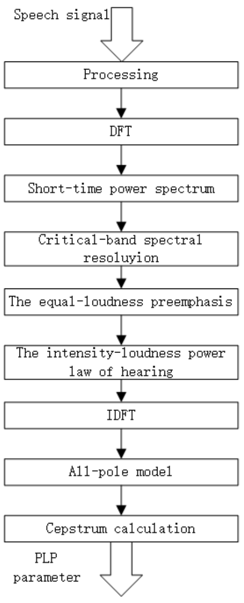
Speech signal
Processing
DFT
Short-time power spectrum
Critical-band spectral resoluyion
The equal-loudness preemphasis
The intensity-loudness power law of hearing
IDFT
All-pole model
Cepstrum calculation
PLP parameter

Fig. 2. PLP feature extraction

2) Spectrum analysis: Cepstrum features contain more information than other parameters and can better represent speech signals. The relationship between cepstrum characteristic $c(n)$ and linear prediction coefficient α_i can be seen in the formula(10).

$$\begin{cases} c(1) = \alpha_1, \\ c(n) = \alpha_n + \sum\limits_{i=1}^{n-1} (1 - i/n)\alpha_i c(n-i), 1 < n \leq p \\ c(n) = \sum\limits_{i=1}^{p} (1 - i/n)\alpha_i c(n-i), n > p \end{cases} \tag{5}$$

In the formula(5), n is cepstrum order and P is linear prediction model order. According to the concept of homomorphic processing and the model of speech signal generation, the cepstrum $c(n)$ of speech signal is equal to the sum of cepstrum of excitation signal and the cepstrum of channel transmission function. The former is widely distributed and can be extended from low time domain to high time domain, while the latter is mainly distributed in low time domain. The speech information carried by speech signal is mainly reflected in the channel transmission function, so in speech recognition, the low-time domain cepstrum of speech signal is usually used to form cepstrum characteristic parameters C.

$$c = [c(1), c(2), ..., c(n)] \tag{6}$$

D. GMM-UBM

GMM is a model composed of multiple Gaussian probability density functions, which is essentially a Likelihood Ratio Detector.

1) Likelihood Ratio Detector: Given a segment of speech, Y , and a hypothesized speaker, S, the task of speaker detection, also referred to as verification, is to determine if Y was spoken by S. An implicit assumption often used is that Y contains speech from only one speaker.

The single-speaker detection task can be restated as a basic hypothesis test between H_0(Y is from the hypothesized speaker S) and H_1(Y is not from the hypothesized speaker S).

The optimum test to decide between these two hypotheses is a likelihood ratio test given by[2]:

$$\frac{p(Y \mid H_0)}{p(Y \mid H_0)} \begin{cases} > \theta, acceptH_0 \\ < \theta, rejectH_0 \end{cases} \tag{7}$$

where $p(Y \mid H_i), i = 0, 1$, is the probability density function for the hypothesis H_i evaluated for the observed speech segment Y , also referred to as the likelihood of the hypothesis H_i given the speech segment. 4 The decision threshold for accepting or rejecting H_0 is θ . The basic goal of a speaker detection system is to determine techniques to compute values for the two likelihoods, $p(Y \mid H_0)$ and $p(Y \mid H_1)$.

2) Universal Background Model(UBM): In the GMM-UBM system we use a single, speaker-independent background model to represent $p(X \mid \lambda_{hyp})$. The UBM is a large GMM trained to represent the speaker-independent distribution of features. We simply merge all the training data to train a UBM.

III. SPEAKER RECOGNITION

The principles of speaker recognition are as follows. VAD processing and MFCC feature extraction are consistent with language recognition.

A. D-vector

1) What is d-vector: Motivated by the powerful feature extraction capability and recent success of deep neural networks (DNNs) applied to speech recognition[7], d-vector propose a speaker recognition technique based on DNN as the speaker feature extractor. The proposed background DNN model for speaker recognition is depicted in Fig.3[4]. The idea is similar to in the sense that neural networks are used to learn speaker specific features. The main differences are that d-vector perform supervised training, and use DNNs instead of convolutional neural networks.

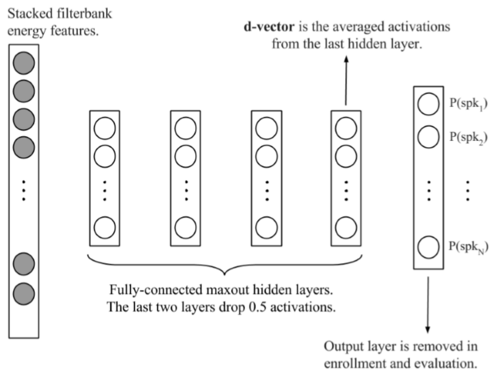
Stacked filterbank energy features.
d-vector is the averaged activations from the last hidden layer.
P(spk$_1$)
P(spk$_2$)
P(spk$_N$)
Fully-connected maxout hidden layers. The last two layers drop 0.5 activations.
Output layer is removed in enrollment and evaluation.

Fig. 3. The background DNN model for speaker recognition

2) DNN extraction: First built a supervised DNN, operating at the frame level, to classify the speakers in the development set. The input of this back ground network is formed by stacking each training frame with its left and right context frames. The number of outputs corresponds to the number of speakers in the development set, N. The target labels are formed as a 1-hot N-dimensional vector where the only non-zero component is the one corresponding to the speaker identity. Fig.3 illustrates the DNN topology.

Once the DNN has been trained successfully, use the accumulated output activations of the last hidden layer as a new speaker representation. That is, for every frame of a given utterance belonging to a new speaker, compute the output activations of the last hidden layer using standard feedforward propagation in the trained DNN, and then accumulate those activations to form a new compact representation of that speaker, the d-vector[4]. Choose to use the output from the last hidden layer instead of the soft-max output layer. First, we can reduce the DNN model size for runtime by pruning away the output layer, and this also enables us to use alarge number of development speakers without increasing DNN size at runtime. Second, we have observed better generalization to unseen speakers from the last hidden layer output. The underlying hypothesis here is that the trained DNN, having learned compact representations of the development set speakers in the output of the last hidden layer. Because it can reduce the DNN model size and observed better generalization to unseen speakers from the last hidden layer output.

3) GRU extraction: We experiment with GRU architectures to extract the d-vector, then mean pool to produce utterance-level speaker embeddings, and train using cross entropy loss based on cosine similarity.

The GRU architecture use recurrent networks for frame-level feature extraction because they have worked well for speech recognition. The details of the proposed GRU architecture are shown in TABLE I $5 * 5$ filter size, $2 * 2$ stride convolution layer reduces dimensionality in both the time and frequency domains, allowing for faster GRU layer computation. Following the convolutional layer are three forward-only GRU layers with 1024 units, recurrent in the time dimension. After the GRU layers, we apply the same average, affine, and length normalization layers. It also uses sequence-wise batch normalization and clipped-ReLU activation in the whole model. The GRU architectures have a number of parameters, 23M-24M, allowing us to better compare their performances.

TABLE I

layer name	struct	stride	dim	param
conv64-s	5*5,64	2*2	2048	6k
GRU	1024 cells	1	1024	9.4M
GRU	1024 cells	1	1024	6.3M
GRU	1024 cells	1	1024	6.3M
average	-	-	1024	0
affine	1024*512	-	512	500K
ln	-	-	512	500K
total				23M

Architecture of the GRU model. “Average” denotes the temporal pooling layer, and “ln” denotes the length normalization layer.

4) LSTM extraction: In this work we use neural networks to obtain the speaker representation of an utterance. The two types of networks are DNN with locally-connected and fully connected layers and a long short-term memory recurrent neural network (LSTM)[8,9] with a single output in Fig.4.

The speaker model is the average over a small number of “enrollment” representations. Use the same network to compute the internal representations of the “test” utterance and of the utterances for the speaker model. To avoid a mismatch, we sample for each training utterance only a few utterances from the same speaker to build the speaker model at training time.

The input of the end-to-end architecture are 1+N utterances, an utterance to be tested and up to N different utterances of the same speaker to estimate the speaker model. To achieve a good tradeoff between data shuffling and memory,

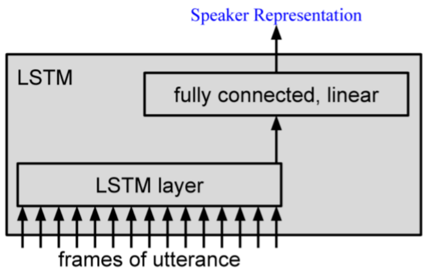

Fig. 4. Long short-term memory recurrent neural network (LSTM) with a single output

the input layer maintains a pool of utterances to sample 1+N utterances from for each training step and gets refreshed frequently for better data shuffling. As a certain number of utterances of the same speaker is needed for the speaker model, the data is presented in small groups of utterances of the same speaker. The end-to-end architecture allows for direct optimization of the evaluation metric using the standard evaluation protocol with consistent speaker models. Conceivably, this will result in better accuracy without the need for heuristics and post processing steps. Moreover, the approach scales well as it neither depends on the number of training speakers and nor requires a minimum number of utterances per speaker.

5) Enrollment and evaluation: Given a set of utterances $X_S = \{O_{S1}, O_{S2}, ..., O_{Sn}\}$ from a speaker s, with observations $O_{Si} = \{o_1, o_2, ..., o_n\}$the process of enrollment can be described as follows. First, use every observation o_j in utterance O_{Si} ,together with its context, to feed the supervised trained DNN. The output of the last hidden layer is then obtained, L2 normalized, and accumulated for all the observations O_j in O_{Si} .We refer to the resulting accumulated vector as the d-vector associated withthe utterance O_{Si} The final representation of the speaker s is derived by averaging all d-vectors corresponding for utterances in X_S.

In this work, we compute the cosine distance[4] between the test d-vector and the claimed speaker's d-vector. We assume test d-vector as vector A and claimed speaker's d-vector as vector B. Then calculate cosine distance θ as follows:

$$\theta = \arccos\left(\frac{A \cdot B}{||A||\ ||B||}\right) = \arccos \frac{\sum\limits_{i=1}^{n} A_i \times B_i}{\sqrt{\sum\limits_{i=1}^{n} (A_i)^2} \times \sqrt{\sum\limits_{i=1}^{n} (B_i)^2}} \tag{8}$$

Then a recognition decision is made by comparing the distance to a threshold.

IV. EXPERIMENT

A. Language recognition

1) Experiment condition: The dataset consist of 2 languages, each consisting of more than 100 speech. In the

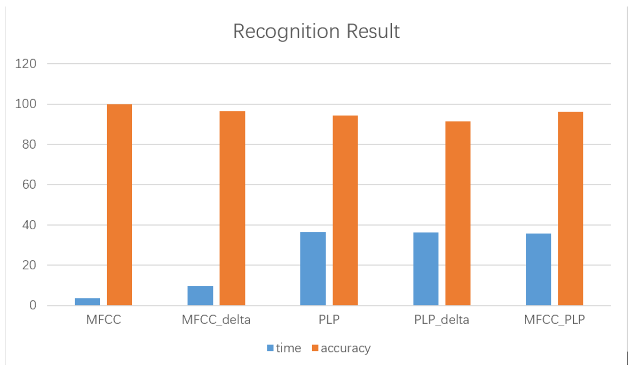

Fig. 5.

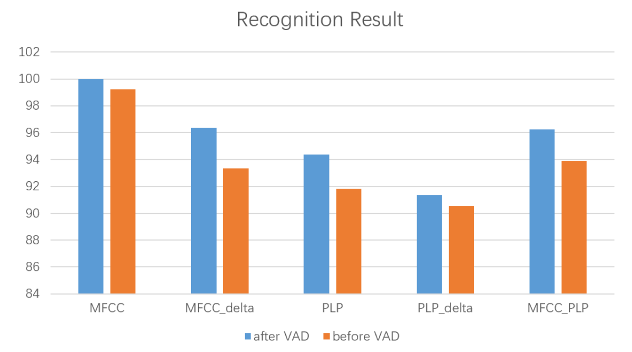

Fig. 6.

test, the voice VAD is processed and each voice is cut into seconds. 30% of the dataset are treated as training sets, and the remaining 70% are treated as test sets.

2) Experiment and result: In order to study the advantages and disadvantages of various features in language recognition, we fixed the classifier GMM and set the order of Gauss parameters (n-components) to 8. Five features were selected, which are MFCC, PLP, MFCC-delta, PLP-delta and MFCC-PLP. Among them, MFCC-delta extracts speech difference features based on MFCC features, PLP-delta extracts speech difference features based on PLP features, and MFCC-PLP is a new feature that combines MFCC features and PLP features of speech.With the above conditions, the test results are as Fig.5.

In order to further study the effects of different features on language recognition, we will study the effects of different amount of dataset and VAD on language recognition results.

We will test the speech before VAD, and compare the results of the previous language recognition, the results show in Fig.6.

According to the results, without VAD, the accuracy of language recognition will decline.By increasing the amount of dataset, the Fig.7 can be obtained compared with the previous results of language recognition.

B. Speaker recognition

1) Vox1 dataset description: VoxCeleb[10] contains over 100,000 utterances for 1,251 celebrities, extracted from videos uploaded to YouTube. The dataset is gender balanced,

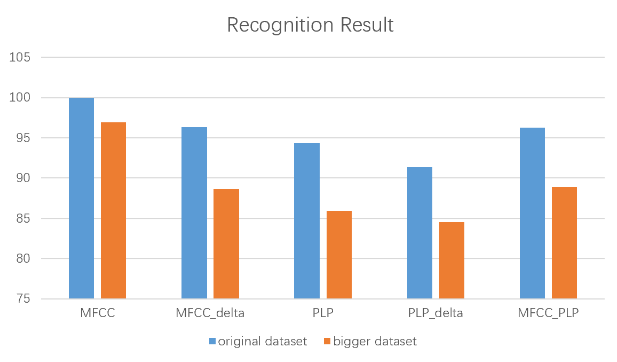

Fig. 7.

with 55% of the speakers male. The speakers span a wide range of different ethnicities, accents, professions and ages. The nationality and gender of each speaker (obtained from Wikipedia) is also provided. Videos included in the dataset are shot in a large number of challenging multi-speaker acoustic environments. These include red carpet, outdoor stadium, quiet studio interviews, speeches given to large audiences, excerpts from professionally shot multimedia, and videos shot on hand-held devices. Crucially, all are degraded with real world noise, consisting of background chatter, laughter, overlapping speech, room acoustics, and there is a range in the quality of recording equipment and channel noise.

There is 1211 celebrities in voxceleb training dataset(about 20G), and 40 celebrities in voxceleb test dataset(about 1G).Limited to the computation ability, we do not use the whole dataset.In training phase, we select 100 celebrities from the voxceleb training dataset as our training dataset. In test phase, we use voxceleb test dataset.And there is no intersection between training dataset and test dataset.After our training dataset and test dataset prepared, we remove the silent speech and cut the speech into segments in seconds.

2) D-vector experiment: We experiment three ways to extract d-vector, which are described above. All the experiments are completed on Google Colab, which provides free Tesla T4 GPU for all of us. We extract mfcc feature from every seconds speech, then the speech are converted to a 98*13 matrix. For DNN model, we flatten the matrix to a 1*1274 vector. For LSTM model, we do nothing. For GRU model, we add a axis to the matrix, so the new matrix is 98*13*1 which like an image.

In training phase, we divide the training dataset into a training dataset and a validation dataset according to a 7:3 ratio.And in test phase, we use part test dataset to generate the d-vector template and use the rest to test the model. The results are as follows:

TABLE II

model	Train acc	Val acc	Train time per epoch(s)	Test acc	Test time(s)
DNN	0.532	0.467	1	0.368	12
GRU	0.939	0.748	99	0.376	70
LSTM	0.788	0.547	29	0.437	50

GRU model is slow,so we set epoch to 30. The training

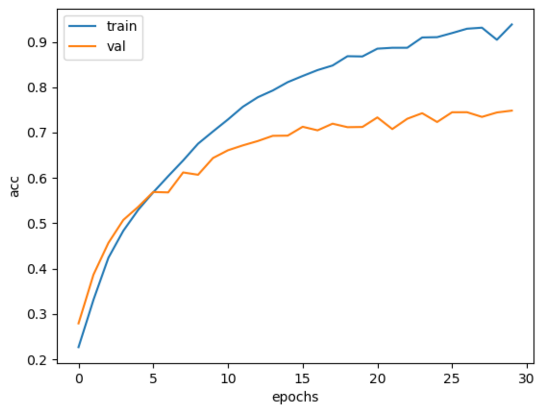

Fig. 8. GRU model's accuracy

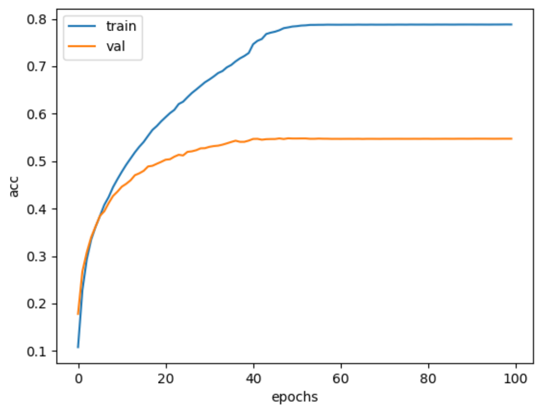

Fig. 10. LSTM model's loss

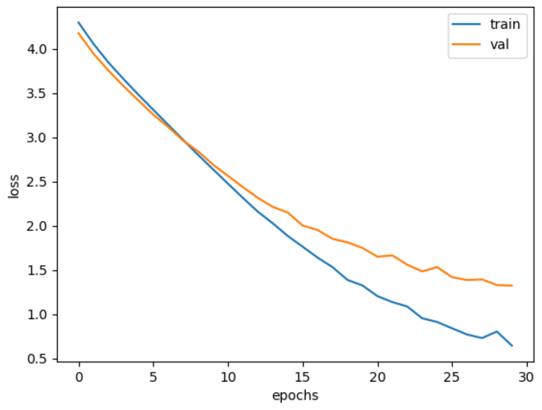

Fig. 9. GRU model's loss

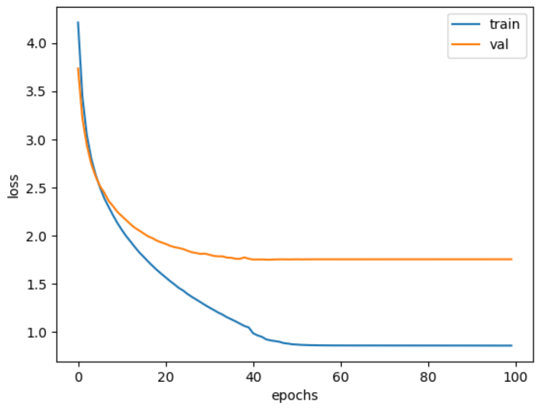

Fig. 11. LSTM model's loss

detail show in Fig.8 and Fig.9.

When training LSTM model, we set epoch to 100. The training log show in Fig.10 and Fig.11. LSTM model has converged when epoch is about 50.

V. RESULT ANALYSIS

According to above, we can conclude that DNN model is faster than GRU and LSTM model,but the accuracy is lower.It make sense that the structure of DNN model is simple and the parameters is less than GRU and LSTM model. GRU model's training accuracy is much higher than test accuracy, so we think GRU model is overfitting. LSTM model achieves the highest test accuracy in three models, and the speed of LSTM model is between DNN model and GRU model.Due to bad results of DNN model, we will discuss GRU model and LSTM model later. GRU model is consist of convolution layer and gru layer, and LSTM model is consist of lstm layer. Both gru layer and lstm layer are categorized as RNN, although their structure is a little different. We are interested in the effect of convolution.But we can not infer any information about the effect of convolution according to above experiments. For more powerful evidence, we do another experiment. We add a convolution layer to LSTM model and other requirement are still the same. The results are show in Fig.12 and Fig.13:

Training is 14s per epoch which is faster than LSTM model and test accuracy is 0.522 which is 0.085 higher than LSTM model. It cost 31s to inference all the test data. According to Fig.8 and Fig.10, we found that they are the same in tendency.

So we can infer that convolution makes great contribution to the model.As is known, convolution network make great progress in computer vision due to its strong power to extract feature from image automatically. In our experiment, it extracts powerful feature from mfcc feature and in the meantime it leads to overfitting. As we mentioned above, GRU model's input is 98*13*1 matrix, which is the same as image, but there is some difference. The pixels in image are equal, but the data in input matrix are not equal(show in Fig.14).Most of the energy is concentrated on the left which is the low frequency part of the voice. Convolution layer can extract pattern in frequency but lstm layer can not. We think that is the reason why convolution has effect on speech.

At last ,we apply d-vector to language identification

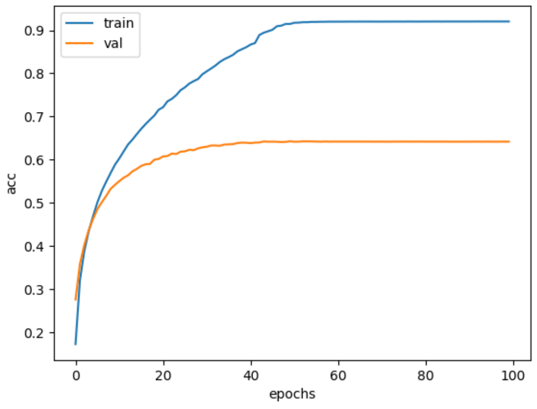

Fig. 12. updated model's loss

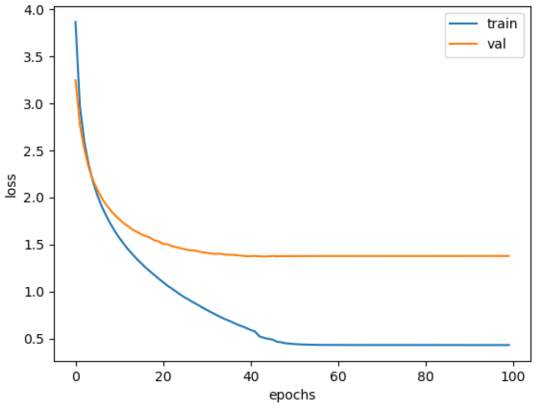

Fig. 13. updated model's loss

model. It achieves 98.1% accuracy and cost 5.5s to inference. So,the generalization ability of the model is good.

VI. CONCLUSION

In language recognition,we apply mfcc and plp feature and GMM-UBM model. we found mfcc and plp feature is still powerful and mfcc feature outperforms plp feature.

In speaker recognition, we explore d-vector based on dnn,gru,lstm model.And d-vector achieve 52.2% accuracy. what's more,we found convolution layer has effect on speech.

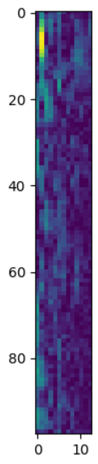

Fig. 14. 98*13 mfcc feature, pixel in the left is bright than the right

REFERENCE

[1] Dalmiya, C.P., Dharun, V.S., Rajesh, and K.P., "An efficient method for Tamil speech recognition using MFCC and DTW for mobile applications," 2013.

[2] Yu-Min Zeng, Zhen-Yang Wu, Falk, T., Chan, W.-Y.. Robust GMM Based Gender Classification using Pitch and RASTA-PLP Parameters of Speech[P]. Machine Learning and Cybernetics, 2006 International Conference on,2006.

[3] Yucesoy, E.,Nabiyev, V.V.. Comparison of MFCC, LPCC and PLP features for the determination of a speaker's gender[P]. Signal Processing and Communications Applications Conference (SIU), 2014 22nd,2014.

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

[5] M.S. Athulya,P.S. Sathidevi. Speaker verification from codec distorted speech for forensic investigation through serial combination of classifiers[J]. Digital Investigation,2018,25.

[6] Mohan, B.J., Babu, and N.R., "Speech recognition using MFCC and DTW," Advances in Electrical Engineering (ICAEE), 2014 International Conference on, 2014.

[7] G.Hinton,L.Deng,D.Yu,G.E.Dahl,A.Mohamed,N.Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B.Kingsbury, "Deepneuralnetworksforacousticmodelingin speech recognition," IEEE Signal Processing Magazine, vol. 29, pp. 82–97, November 2012.

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

[9] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in Interspeech, Singapore, Sep. 2014.

[10] A. Nagrani, J. Chung, and A. Zisserman, "VoxCeleb: A large-scale speaker identification dataset," in Interspeech, 2017, pp. 2616–2620.

[11] P. Kenny, P. Ouellet, N. Dehak, V. Gupta, and P. Dumouchel, "A study of interspeaker variability in speaker verfication," IEEE Transactions on Audio, Speech, and Language Processing, vol. 16, pp. 980–988, 2008.

[12] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker vertfication," IEEE TransactionsonAudio,Speech,andLanguageProcessing,vol. 19, pp. 788–798, 2011.