Speech analysis & contrast system

REPORT for speech recognition course design.

Group member:

1953348 叶栩冰　1953066 刘昕宇　1953196 张铃沛

Faculty advisor:

沈莹

2021/12/31

Speech analysis & contrast system

Abstract

This project for Tongji university 2021 fall semester speech recognition course project, project for the theme of speech analysis system, can the content of the speech for speech for input, speech emotion, voiceprint gender, basic information, such as the volume voice voice print characters matching analysis and comparison, and presents a simple interactive interface and the result analysis.

1. Problems & difficulties

Our project is mainly divided into the following three modules: Chinese speech recognition, speech emotion and gender recognition, and voice print recognition and comparison. The follow-up questions will be divided into three parts.

1.1 Chinese speech recognition

The function of this module is relatively single, but the realization is relatively complicated. After the user inputs a piece of audio, according to the trained model, the audio is converted into text and displayed in front of the user. There are two difficulties, one is to obtain the audio signal of a single word from continuous audio, and the other is to select suitable words from the vocabulary to form a sentence based on a series of discrete sounds. The model trained by the system is the conversion of speech to language, that is, input a piece of audio, and after operations such as feature extraction and word segmentation, a language sequence is finally obtained.

1.2 Speech emotion and gender recognition

This module is to solve the problem of emotion recognition of audio. The specific description is as follows: Firstly, define the labels of two dimensions, namely gender (male and female) and emotion (angry, happy, SAD, calm and Fear). After we input an audio clip, we need to analyze the characteristics of the audio clip to predict which gender the audio comes from and which emotion it belongs to.

The complexity of emotion itself makes it very difficult to collect and sort out emotion speech data, which makes it difficult to obtain high-quality emotion corpus. For discrete emotion speech database, it is difficult to obtain speech that meets the requirement of corpus naturalness and emotion purity.

For the establishment of dimensional affective speech database, the difficulty lies not in the acquisition of corpus, but in the collation of corpus and the annotation of emotion. It is very difficult to label emotion in a large number of speech data sets.

The relationship between emotion and acoustic features: The initial difference between computer and human emotion recognition mechanism is the extraction of acoustic features related to emotion and the determination of the correlation between emotion and acoustic features. Therefore, if the computer can't accurately or as close as possible to the human way of emotional speech acoustic feature extraction and the correct association and mapping, will make computer speech emotion recognition system is established on the basis of a deviation from the actual, which leads to the following recognition mechanism and the gap between the human brain processing mechanism is more and more big, can't achieve the desired effect. At present, MFCCS is a good feature extraction method. At the same time, how to define the optimal extraction time of emotional acoustic features or fuse acoustic features of different time lengths also needs to be considered.

1.3 Voice print recognition and comparison

This module is a voiceprint comparison system, that is to analyze the voiceprint of the two ends of the input voice, judge whether they are the same person's voice, and give the degree of similarity between the two voices. Our initial idea was to build a deep learning model to predict the voiceprint. However, when testing, we found that the model could not predict the voiceprint outside the training set. Therefore, we simply modified the model to calculate the voice print similarity without predicting the voice print's ownership. These two models are explained and analyzed in the following part.

The main difficulty of voiceprint comparison system lies in the construction of the prediction model and the derivation of the feature matrix and the calculation of its similarity.

2. Survey on topic & used methods

The team members did a full research on the topic and selected different solutions to the three main problems. This part briefly describes the data set, data processing, model selection, loss function and other main model construction methods used, and analyzes and compares the effects of the model and each method selection process.

2.1 Chinese speech recognition

2.1.1 Dataset

- dataset-free_st_chinese_mandarin_corpus

  The free Chinese corpus provided by Surfingtech (www.surfing.ai) contains the voices of 855 speakers and 102,600 sentences. We compared three different data sets and found that the effect of free-st is relatively best. , There is a high probability that the entire sentence input by the user can be recognized, the error rate is low, and the adaptability to long speech is also relatively good. Here are two other data sets we tested

- dataset-thchs30

  THCHS-30 was recorded through a single carbon microphone in a quiet office environment, with a total duration of more than 30 hours. Most of the people involved in the recording are college students who can speak fluent Mandarin. The sampling frequency is 16kHz, and the sampling size is 16bits. It comes from Tsinghua University. The recorded content is mainly text or news. It is one of the open source libraries for Mandarin recognition in China.

- dataset-aishell

  Hill Shell's Mandarin Chinese voice database, text design, smart home, driverless, industrial production and other fields. Use microphones and mobile phones to record, and 400 speakers from different accent regions participated in the recording. After professional phonetic proofreaders write annotations. A total of 178 hours, 340 people in the training set, 20 people in the test set, and 40 people in the verification set. It is also one of the domestic open source libraries for Mandarin recognition.

After comparison, we finally pickfree_st_chinese_mandarin_corpus.

2.1.2 Data handling

After segmenting the audio, the features are extracted, and the label of the voice is combined with the voice data itself, which is convenient for calling during training.

First, sort the audio and put the short audio before the long audio, so that the system can give priority to training short audio. Training very deep networks from scratch (or rnn with many steps) may fail early in training because the output and radiation must be propagated through many untuned weight layers. Except for the explosion gradient. CTC usually ends up assigning a probability close to zero for very long transcriptions, which makes the gradient descent quite unstable. We use a course learning strategy called SortaGrad: We use the length of the utterance as a heuristic for difficulty, and train shorter (easier) utterances first.

Specifically, in the first training stage, we iterate the mini-batch in the training set in increasing order of the longest utterance length in the mini-batch. After the first epoch, training returns to random order in mini-batch. SortaGrad improves the stability of training and reduces training costs

2.1.3 Model building

Use RNN recurrent neural network to build the model. The purpose of RNN is to process sequence data. In the traditional neural network model, from the input layer to the hidden layer and then to the output layer, the layers are fully connected, and the nodes between each layer are not connected. We predict what the next word of the sentence will be, and generally we need to use the previous word. RNNs are called recurrent neural networks, that is, the current output of a sequence is also related to the previous output. The specific form of expression is that the network will memorize the previous information and apply it to the calculation of the current output, that is, the nodes between the hidden layers are no longer unconnected but connected, and the input of the hidden layer not only includes the output of the input layer It also includes the output of the hidden layer at the previous moment. In theory, RNNs can process sequence data of any length.

In our network, there are a total of multiple convolutional input layers, plus a multi-layer RNN recurrent neural network for training. The realization of a recurrent layer is:

$$h_t^l = f(W^l h_t^{l-1} + U^l h_{t-1}^l + b)$$

Among them, the activation amount of layer l at time step t is determined by the activation amount of the previous layer h_t^{l-1} at the same time step t and the current layer's activation amount at the previous time step h_{t-1}^l activation amount combined with calculation.

In order to effectively absorb data when expanding the training set, we increase the depth of the network by adding more loop layers. However, as the network size and depth increase, the use of gradient descent to train the network is prone to problems. We used the BatchNorm method to train deeper nets faster. Research shows that BatchNorm can speed up the convergence speed of Rnn training. When applied to a very deep network of Rnn on a large data set, BN greatly improves the final generalization error while accelerating training.

For each hidden unit, we calculate the mean and variance statistics of the sequence length of all items in the mini-batch to achieve batch standardization:

$$h_t^l = f(B(W^l h_t^{l-1}) + U^l h_{t-1}^l)$$

2.1.4 Loss function-CTC

The CTC loss function gives the output distribution of all possible Y for a given X. You can use this distribution to infer a possible output or estimate the probability of a given output. Not all methods of calculating loss functions and performing reasoning are easy to handle. For a given input, we want to train our model to maximize the probability of assigning it to the correct answer.

Its basic idea is that its basic idea is to interpret the network output as a probability distribution of the overall possible label sequence, and the condition is a given input sequence. Given this distribution, an objective function can be derived to directly maximize the probability of the correct label.

It is found that the use of CTC is very unstable in the early stage of training, which affects the training of the final model. This is because training a very deep network from scratch (or Rnn with many steps) may fail early in the training, because the output and gradient must be propagated through many untuned weight layers. Except for the explosion gradient. CTC usually finally assigns a probability of close to zero for very long transcriptions, which makes the gradient descent quite unstable, which is one of the reasons for the introduction of sortagrad.

2.1.5 Data enhance-SpecAugment

Since the training audio data selected by us are some audio data with standard pronunciation and low ambient noise, the trained model may be unable to resist some audio mutation and noise, so we select SpecAugment as the data enhancement method. SpecAugment modifies the spectral diagram by distorting the time domain signal, masking the frequency domain channel, and masking the time domain channel. This enhancement method can be used to increase the robustness of the network, to resist deformation in the time domain and partial fragment loss in the frequency domain. The sound waves are transformed into spectral maps before being fed into the network as training data. Data enhancement is usually first applied to sound waves and then transformed into spectral images. For each iteration, the new sound, enhanced by the data, needs to be converted into a spectral map. In our approach, we augment the spectrum directly, rather than the acoustic data. Because the data enhancement of this method is directly used on the input features, it can be added dynamically in real time, without the need for a lot of computational costs that affect the training speed like the data enhancement of sound waves.

2.2 Speech emotion and gender recognition

2.2.1 Dataset-RAVDSS

Due to the above mentioned "dimension of emotional speech database established" the difficulty, it is difficult for us to personally to arrange and emotion tagging corpus - because in this case, the individual subjective feelings may lead to emotional understanding deviation, caused from the training models and can't get accurate results (most people recognized the results). So we went through a lot of stuff on the Internet and finally found the RAVDESS English dataset that we are using now.

RAVDESS contains 7,356 files. The database consisted of 24 professional actors (12 women, 12 men) speaking two word-matching statements in a neutral North American accent. The speech included expressions of calm, happiness, sadness, anger, fear, surprise and disgust, each with two emotional levels (normal and strong), plus a neutral expression. In this project, we did not use the entire data set, but extracted five categories of calm, happiness, sadness, anger and fear, with an additional dimension of gender.

2.2.2 Data processing

1. Pre-emphasis, frame and window the speech first;
2. For each analysis window, obtain the corresponding spectrum through STFT;

   $$X(n,\omega)=\sum_{m=-\infty}^{\infty}x(m)w(n-m)e^{-j\omega m}$$

   Where $x(m)$ is the input signal, $w(m)$is the window function, which is reversed in time and has an offset of n samples. $X(n,\omega)$ is time n and ω frequency! is a two-dimensional function that connects the time domain and frequency domain of the signal. We You can perform time-frequency analysis on the signal based on this, for example$S(n,\omega)=|X(n,\omega)|^2$ is the so-called spectrogram of the speech signal.
3. Pass the Mel filter bank on the above spectrum to get the Mel spectrum;

4. Perform analysis on the Mel spectrum (take the logarithm and do the inverse transformation, the actual inverse transformation is generally realized by DCT discrete cosine transform, and take the 2nd to 13th coefficients after DCT as the MFCC coefficients), plus The energy becomes 13-dimensional, and the MFCC is obtained. This MFCC is the characteristic of this frame of speech;

   In this way, we process and crop each piece of audio to obtain a 216*13 MFCC feature vector.

2.2.3 Model specification-Cnn1D

The project selected the cnn1D one-dimensional convolutional neural network for prediction, and the detailed network architecture will not be analyzed here. The general process is: averaging the feature vector of each processed audio on the 13th dimension to obtain a 216*1 vector, and use this as an input to train the model. After each round of training, an evaluation is performed to calculate the accuracy of the model to observe the convergence of the model. When the training is all completed, save the trained model for evaluation of large amounts of data and test of a single audio.

As to the activation function, we chose pick ReLU as the activation function of the hidden layer and softmax as the activation function of the output layer for the reason that this is essentially a multi-classification problem.

2.3 Voice print recognition and comparison

2.3.1 Dataset-Zhvoice

The data set used in this module is Zhvoice Chinese corpus, which has a total of 3242 individual voice data, including 1,130,000 + voice data. Compared with other data sets, this data set labels tasks and has a large amount of data, and the number of characters can meet the training requirements of the model.

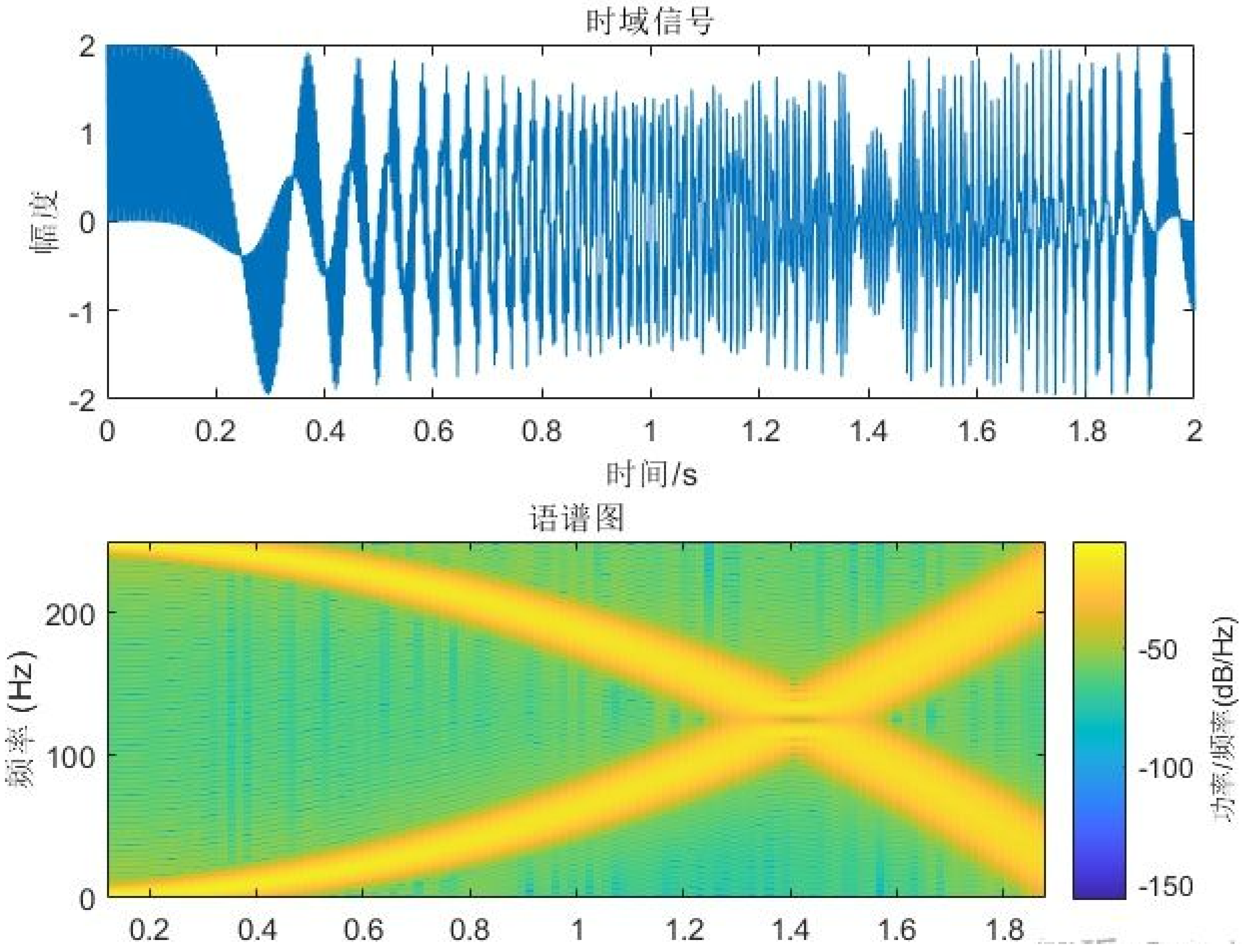
2.3.2 Data Processing-STFT

Short-Time Fourier Transform, STFT, which is defined as:

$$X(n,\omega)=\sum_{m=-\infty}^{\infty}x(m)w(n-m)e^{-j\omega m}$$

Among the formula, $x(m)$ is the input signal， $w(m)$ is the window function，it reverses in time and has an offset of n samples. $X(n,\omega)$ is a two-dimensional function of time n and frequency ω , it connects the time domain and frequency domain of the signal, and we can analyze the time frequency of the signal, for instance, $S(n,\omega)=|X(n,\omega)|^2$ is a speech signal called a spectrogram.

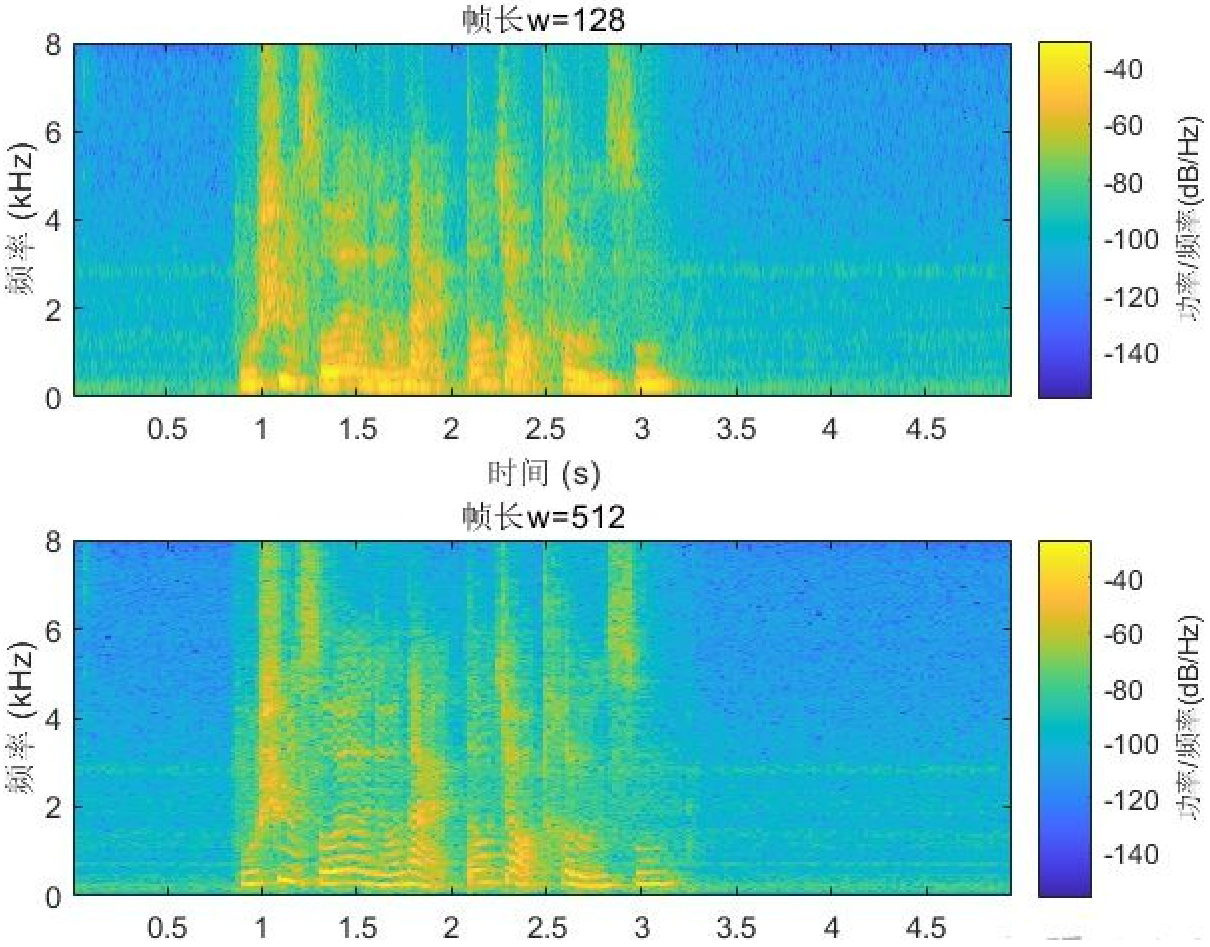
The following figure shows the spectrogram of two sweep signals after superposition.



It can be seen that the signal is the superposition of a 0-250Hz twice increasing sweep signal and a 250-0Hz twice decreasing sweep signal. By STFT, we can easily obtain the time-varying characteristics of non-stationary signals.

Different window lengths are used in spectral calculation $S(n,\omega)$, Two kinds of spectrograms, narrowband and wideband, can be obtained. Long time Windows (at least two pitch periods) are often used to calculate narrowband spectrograms, while short Windows are used to calculate broad-band spectrograms. Narrowband speech spectrogram has high frequency resolution and low time resolution. With good frequency resolution, each harmonic component of speech can be more easily identified and displayed as horizontal fringe on speech spectrogram. On the contrary, broadband spectrogram has high time resolution and low frequency resolution. Low frequency resolution can only obtain spectral envelope, and good time resolution is suitable for analyzing and testing Chinese pronunciation.

As shown below, the spectrograms of a speech with frame length 128 and 512 are respectively.



It can be seen that for the STFT with fixed frame length, the time resolution and frequency resolution are fixed in the global scope.

We then processed the audio data set, converting the audio to a 257*257 STFT amplitude spectrum.

2.3.3 Model specification-Resnet

Resnet-50 residual neural network is selected for prediction in the project. Detailed network architecture is not analyzed here.

At the end of each training round, model evaluation was performed to calculate the accuracy of the model and observe the convergence of the model. Similarly, the model is saved once at the end of each round of training, and the model parameters that can be restored to training are saved respectively, which can also be used as pre-training model parameters. The prediction model is also saved for later prediction.

2.3.4 Loss function-ArcFace

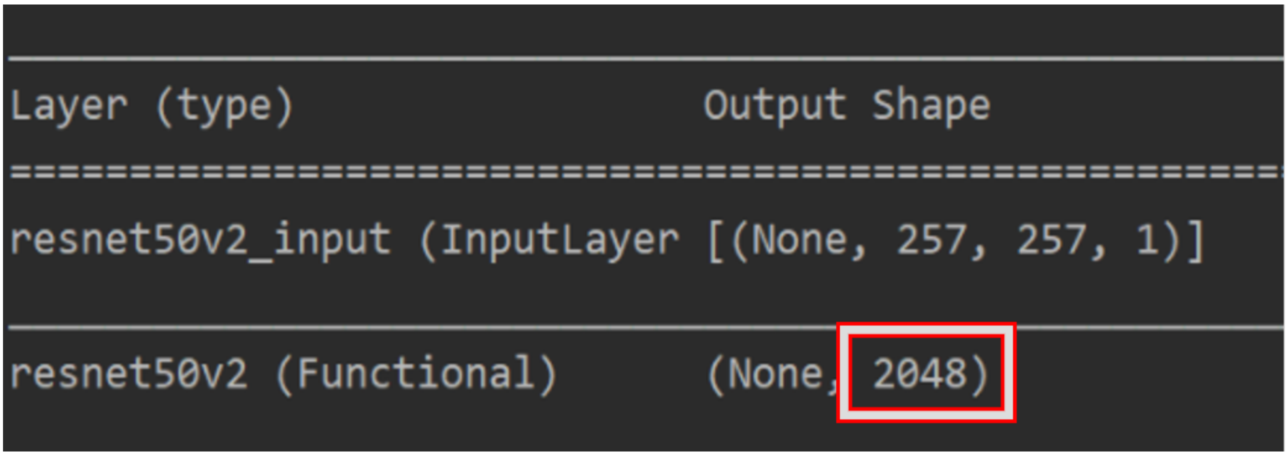
ArcFace is published by Imperial College London in January 2018. Based on Sphere Face, it improves the normalization and additive Angle interval of feature vectors, improves the separability between classes, and strengthens the intra-class compactness and differences between classes. ArcFace has the advantages of high performance, easy programming, low complexity and high training efficiency. For the normalization of eigenvectors and weights, for θ plus the Angle interval m, the Angle interval has more direct effect on angles than the cosine interval.

As the embedding features are distributed around each feature Centre on the hypersphere, we add an additive angular margin penalty m between xi and Wyi to simultaneously enhance the intra-class compactness and inter-class discrepancy. Since the proposed additive angular margin penalty is equal to the geodesic distance margin penalty in the normalized hypersphere, we name our method as ArcFace.

$$L_3 = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s(\cos(\theta_{y_i}+m))}}{e^{s(\cos(\theta_{y_i}+m))}+\sum_{j=1,j\neq y_i}^{n}e^{s\cos\theta_j}}. \quad (3)$$

2.3.5 Refactoring model-Diagonal cosine

In the problem brief and difficult parts, we found the following problems: Due to the limitation of training data set, we could not compare our own voice with more diverse voices, and we could not give the similar relationship between the two voices. Therefore, we reconstructed the model. Instead of predicting the input speech, we compared the output eigenvalues in the process of the model.



```
________________________________________________________________
Layer (type)                 Output Shape
================================================================
resnet50v2_input (InputLayer [(None, 257, 257, 1)]

________________________________________________________________
resnet50v2 (Functional)      (None, 2048)
```

That is, in the feature matrix with the output dimension of (2048, 1) in the residual neural network, we only need to solve the diagonal cosine value of the two input speech feature values to the similarity, and then set the threshold value to judge whether the two audio is from the same person.

$$dist = \frac{np.dot(feature1, feature2)}{np.linalg.norm(feature1) * np.linalg.norm(feature2)}$$

3. Workflow & functionality of program

This part will combine the main functions in the code, briefly describe the model architecture and workflow, each model will be analyzed and explained from data construction, data processing to model training.

3.1 Chinese speech recognition

3.1.1 Creating data & handling data

We should build a data generator for training data and testing data

```
train_generator = DataGenerator(vocab_filepath=args.vocab_path,

 mean_std_filepath=args.mean_std_path,

 augmentation_config=augmentation_config,
                                max_duration=args.max_duration,
                                min_duration=args.min_duration,
                                place=place)
```

The parameters are passed through the format of the configuration file. The parameters of the generator are presets for the generator, including the input vocabulary, the file of the average standard value of the data set, and some data operations. Then call the generator function to generate sub-batch training data and test data

```
train_batch_reader =
train_generator.batch_reader_creator(manifest_path=args.train_manif
est,
batch_size=args.batch_size,
             shuffle_method=args.shuffle_method)
```

In the data generator, the audio address and the audio corresponding text data are connected together to form a tuple

```
inst = self.process_utterance(instance["audio_filepath"],
instance["text"])
```

Information reading uses the fluid part of paddle to use its set_batch_generator function to pass in the data reader.

3.1.2 Model training

Use paddle for training, call paddle.static.Executor to generate a model trainer, and initialize it

```
exe = paddle.static.Executor(self._place)
exe.run(startup_prog)
```

Then the training is performed and the training process data is saved

```
fetch = exe.run(program=train_compiled_prog, fetch_list=
[ctc_loss.name], return_numpy=False)
```

The returned FETCH contains the loss rate of training. After calculation, the loss of an EPOCH can be obtained and the time of a training can be calculated. The model was saved once after every 10,000 batches and once after each epoch until the end of the training. These are main params of the model.

PARAMS	NUM
epoch_num	50
batch_size	16
max_duration	20.0
min_duration	0.5
num_Rnn_layer	3
num_conv1_layer	2
initial_learning_rate	5e-4

3.1.3 Model Predict

When the speech prediction is needed, the audio file is loaded and preprocessed, including extracting audio features and obtaining some audio parameters

```
audio_feature = self.audio_process.process_utterance(audio_path)
audio_len = audio_feature.shape[1]
mask_shape0 = (audio_feature.shape[0] - 1) // 2 + 1
mask_shape1 = (audio_feature.shape[1] - 1) // 3 + 1
mask_max_len = (audio_len - 1) // 3 + 1
mask_ones = np.ones((mask_shape0, mask_shape1))
mask_zeros = np.zeros((mask_shape0, mask_max_len - mask_shape1))
mask = np.repeat(np.reshape(np.concatenate((mask_ones, mask_zeros),
axis=1),
                            (1, mask_shape0, mask_max_len)), 32,
axis=0)
```

The Paddleinfer paddle will then be used to generate Predictor and call it to predict the model

Once the data is bound, call Predictor's Run method

```
self.predictor.run()
```

You can then get the output

```
output_handle =
self.predictor.get_output_handle(self.output_names[0])
```

The output that is returned here is a two-dimensional array and the predicted result is a two-dimensional sequence of probabilities, each row being the probability of each word that the current sound is likely to pick up. Take the column coordinates of the word with the highest probability in each row and form a one-dimensional list, called the maximum index of each row. Then take out the probability values corresponding to all the maximum indexes in the whole probability sequence table to form a one-dimensional list, which is called the maximum probability list. Process to remove empty indexes and consecutive identical indexes from the maximum index list (which removes most of the data). According to the ultimate maximum index table to the word dictionary to find the corresponding words of each index, connected together to form a sentence. Adding up all the maximum probabilities and dividing by the length of the maximum probability list gives the average accuracy per word of the sentence. This accuracy rate will be returned to the front end as the score of the speech prediction.

3.2 Speech emotion and gender recognition

3.2.1 Create data

We use the data set to create a Dataframe, where the format of the Dataframe is `<speech file path\speech classification label>`. The purpose of creating this list is to facilitate the reading of files during data preprocessing. Among them, our data set is an audio file in .wav format that has been labeled. The voice classification label refers to the five emotional dimensions of "angry", "sad", "happy", "calm" and "fear". And "male" and "female" two gender dimension labels. In the specific implementation process, in order to simplify the classification difficulty and reduce the training time, we combined gender and emotion into one dimension, so that we converted it into a multi-classification problem of 10 categories.

```
feeling_list=[]
for item in mylist:
    if item[6:-16]=='02' and int(item[18:-4])%2==0:
        feeling_list.append('female_calm')
    elif item[6:-16]=='02' and int(item[18:-4])%2==1:
        feeling_list.append('male_calm')
    elif item[6:-16]=='03' and int(item[18:-4])%2==0:
        feeling_list.append('female_happy')
    elif item[6:-16]=='03' and int(item[18:-4])%2==1:
        feeling_list.append('male_happy')
    elif item[6:-16]=='04' and int(item[18:-4])%2==0;
......
labels = pd.DataFrame(feeling_list)
```

As shown in the figure above, the data is labeled in extract.py. After creating the DataFrame, we check the generated list file and modify the entries that caused the error, and finally get the following list containing the file path and the corresponding label.

```
,path,label
0,test_list\female\03-01-02-01-01-01-02.wav,1
1,test_list\female\03-01-02-01-01-01-04.wav,1
2,test_list\female\03-01-02-01-01-01-06.wav,1
3,test_list\female\03-01-02-01-01-01-08.wav,1
...
```

3.2.2 Data processing

After extract.py completes the creation of the data, we start to extract MFCC features from all the audio files in the training set, and store the resulting feature vectors in a .csv file to prepare for subsequent training.

```
df = pd.DataFrame(columns=['feature'])
bookmark=0
for index,y in enumerate(mylist):
    if mylist[index][6:-16]!='01' and mylist[index][6:-16]!='07'
and mylist[index][6:-16]!='08' and mylist[index][:2]!='su' and
mylist[index][:1]!='n' and mylist[index][:1]!='d':
        X, sample_rate = librosa.load('RawData/'+y,
res_type='kaiser_fast',duration=2.5,sr=22050*2,offset=0.5)
        sample_rate = np.array(sample_rate)
        mfccs = np.mean(librosa.feature.mfcc(y=X,
                                            sr=sample_rate,
                                            n_mfcc=13),
                        axis=0)
        feature = mfccs
```

3.2.3 Model training

Run train.py to start training the model, use a one-dimensional convolution model, and set the data input layer to [None, 216, 1]. In order to observe the convergence of the model and the prediction accuracy, we re-run the data after each epoch Chaos and evaluate. And after the training is completed, the model is saved as a .h5 file, which will be used for the evaluation of a large amount of test data and the prediction of a single voice.

```
model = Sequential()

model.add(Conv1D(256, 5,padding='same',
                 input_shape=(216,1)))
model.add(Activation('relu'))
model.add(Conv1D(128, 5,padding='same'))
......
model.add(Activation('relu'))
model.add(Flatten())
model.add(Dense(10))
model.add(Activation('softmax'))
opt = keras.optimizers.rmsprop(lr=0.00001, decay=1e-6)
```

```
model.summary()
model.compile(loss='categorical_crossentropy',
optimizer=opt,metrics=['accuracy'])
```

We remove the entire training part to avoid unnecessary long training.

```
cnnhistory=model.fit(x_traincnn, y_train, batch_size=16,
epochs=700, validation_data=(x_testcnn, y_test))
```

These are some parameters of the model.

PARAMS	NUM
epoch_num	60
batch_size	32
input_shape	(216, 1)
output_size	10
initial_learning_rate	1e-3

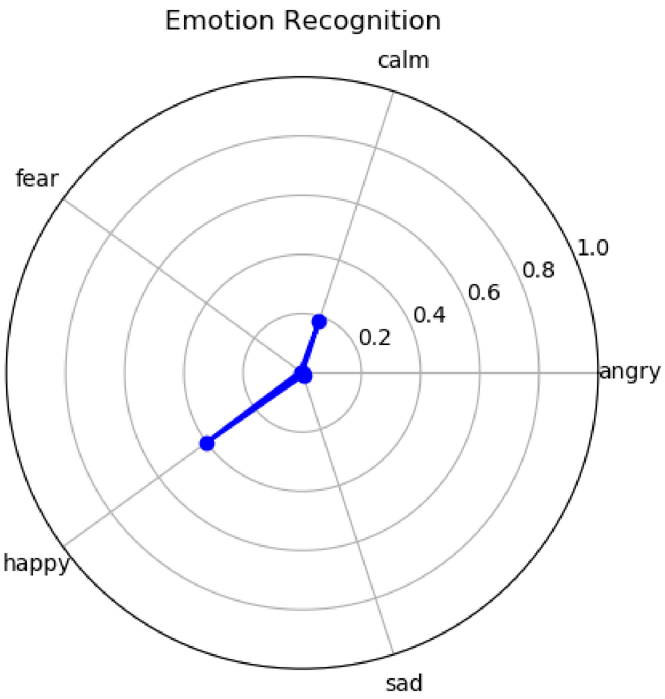
3.2.4 Predict emotion with model

Run the prediction function predict.py, first extract the MFCC features of the input audio and convert them to the required dimensions, and then extract the trained model. The trained model is used to predict the input audio, and the probability vector of the audio on each label is obtained. The sum of all the components in the probability vector is 1, and finding the label corresponding to the largest component in the vector is the result of our prediction. We show the prediction results through the radar chart, which is more clear.

```
def predict(wav_file, loaded_model):
    X, sample_rate = librosa.load(wav_file, res_type='kaiser_fast',
duration=2.5, sr=22050 * 2, offset=0.5)
......
    livepreds = loaded_model.predict(twodim,
                                     batch_size=32,
                                     verbose=1)
    return livepreds
```

```
def Judge_render(livepreds):
    ifmale = 0
    iffemale = 0
    livepreds = livepreds.reshape(10, )
    for i in range(5):
        iffemale = iffemale + livepreds[i]
    for j in range(5, 10):
        ifmale = ifmale + livepreds[j]
    if ifmale > iffemale:
        return "male", livepreds[5:10]
    else:
        return "female", livepreds[0:5]
```

We obtain radar graphs and weight matrices for emotion and gender recognition results.



```
[[2.1368088e-01 4.1267309e-02 1.4035265e-07 3.3199336e-02
1.1228149e-01
3.0333758e-05 1.8161860e-01 7.8319320e-03 4.0111807e-01 8.9718904e-
03]]
male
```

3.3 Voice print recognition and comparison

3.3.1 Data creating

We will use the data set to create a data list, the format of the data list is`< speech file path \t speech classification label >`, the creation of this list is mainly for the convenience of reading later, but also for the convenience of reading using other speech data sets, speech classification label refers to the unique ID of the speaker, different speech data sets, we can write these datasets in the same data list by writing the corresponding functions that generate the data list.

Data label processing in the create_data.py, because mp3 format audio reading speed is slow, so to convert all mp3 format audio to WAV format, after creating the data list, some data may be wrong, so we want to check, delete the wrong data. Perform the following procedure to complete the data preparation. The following data formats and labels are formed.

```
Speech-Recognition-Final-Project/5_895/5_895_20170614203758.wav
3238
Speech-Recognition-Final-Project/5_895/5_895_20170614214007.wav
3238
Speech-Recognition-Final-Project/5_941/5_941_20170613151344.wav
3239
Speech-Recognition-Final-Project/5_941/5_941_20170614221329.wav
3239
Speech-Recognition-Final-Project/5_941/5_941_20170616153308.wav
3239
Speech-Recognition-Final-Project/5_968/5_968_20170614162657.wav
3240
```

3.3.2 Data processing

With the data list and mean standard values created above, it can be used for training reads. The speech data is mainly converted into the amplitude spectrum of short-time Fourier transform, and data enhancement is carried out in this step, such as random turnover splicing, random clipping. After processing, we get a 257 by 257 STFT amplitude spectrum.

```
# STFT
wav, sr_ret = librosa.load(audio_path, sr=sr)
linear = librosa.stft(extended_wav, n_fft=n_fft,
win_length=win_length, hop_length=hop_length)
linear_T = linear.T
mag, _ = librosa.magphase(linear_T)
mag_T = mag.T
```

3.3.3 Model training

Run train.py to start training the model, using RESnet50 model oftensorflow, and set the data input layer to [None, 1, 257, 257], that is, the shape of the amplitude spectrum of the STFT. In order to better observe the training effect of the model and save the training time, model evaluation was performed after each training round to calculate the accuracy of the model and observe the convergence of the model. Similarly, the model is saved once at the end of each round of training, and the model parameters that can be restored to training are saved respectively, which can also be used as pre-training model parameters.

```
def create_model(input_shape):
    # Build model
    model = tf.keras.Sequential()
    model.add(ResNet50V2(input_shape=input_shape,
include_top=False, weights=None, pooling='max'))
    model.add(BatchNormalization())
    model.add(Dense(units=512,
kernel_regularizer=tf.keras.regularizers.l2(5e-4),
name='feature_output'))
    model.add(ArcNet(num_classes=args.num_classes))
    return model
```

These are some parameters of the model.

PARAMS	NUM
epoch_num	50
batch_size	16
input_shape	(257, 257, 1)
output_size	(3242, 1)
ResNet50V2	1

PARAMS	NUM
feature_used_shape	(2048, 1)
initial_learning_rate	1e-3

This is the framework of the model.

```
Model: "Resnet-50"
_________________________________________________________________
Layer (type)                 Output Shape             Param #
=================================================================
resnet50v2_input (InputLayer [(None, 257, 257, 1)]    0
_________________________________________________________________
resnet50v2 (Functional)      (None, 2048)             23558528
_________________________________________________________________
batch_normalization (BatchNo (None, 2048)             8192
=================================================================
Total params: 23,566,720
Trainable params: 23,517,184
Non-trainable params: 49,536
```

3.3.4 Voice print contrast

Run voiceprint_predict.py, ssing the audio eigenvalue output by residual neural network, their diagonal cosine value is solved, and the result is used as their acquaintance degree.

```
# Voice print contrast
    feature1 = infer(args.audio_path1, model, input_shape)[0]
    feature2 = infer(args.audio_path2, model, input_shape)[0]
    # acquaintance degree
    cos = np.dot(feature1, feature2) / (np.linalg.norm(feature1) *
np.linalg.norm(feature2))
    if dist > 0.7:
        print("为同一个人，相似度为：%f" % (cos))
        return True, dist
    else:
        print("不是同一个人，相似度为：%f" % (cos))
        return False, dist
```

4.Performance evaluation & result interpretation

We analyzed the results of the constructed model, and calculated the accuracy rate, regression rate, F1-score, ROC and AUC parameters of the model. Different types of models may have different evaluated values due to different output forms. In addition, we will use our own test audio in this module to test and show test samples.

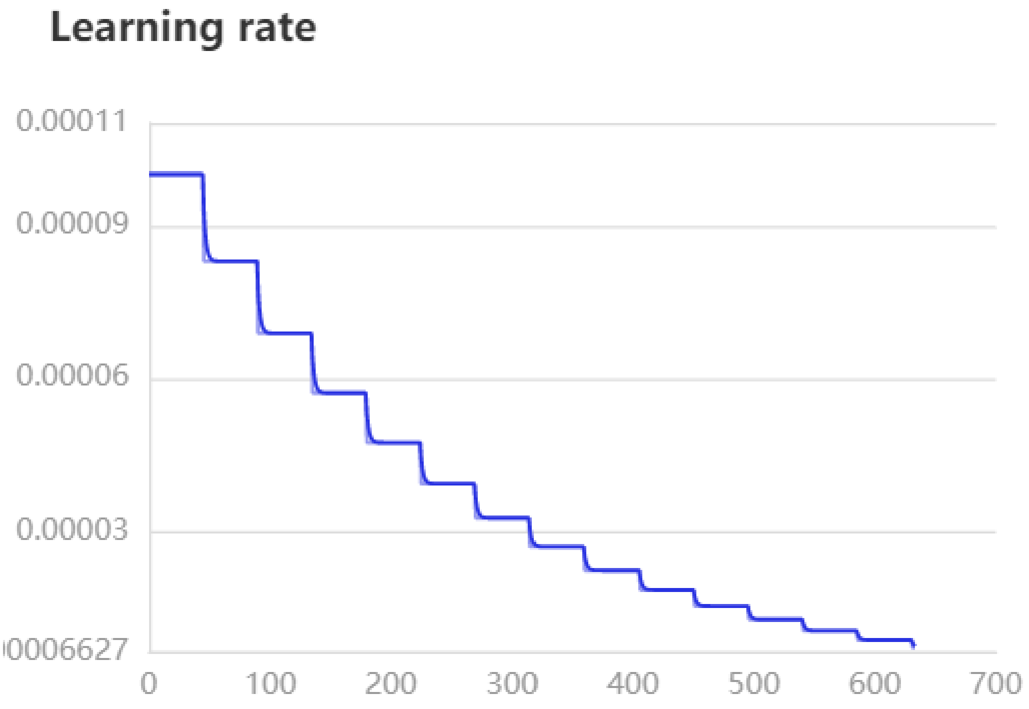
4.1 Chinese speech recognition

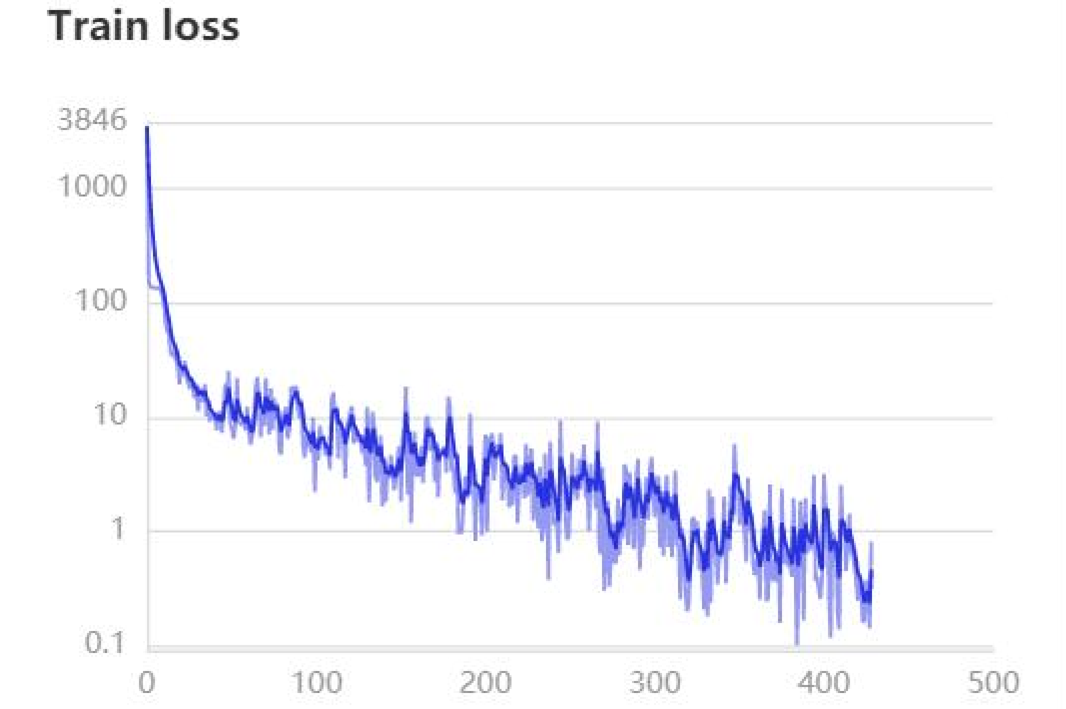
We used all three data sets for training, and evaluated and analyzed the model as follows:

Compared three models, thchs30 recognition effect is centered, around 70%, the correct and audio accuracy is better, but the pronunciation for words some relatively weak, may be due to the tsinghua university library contains a lot of proper nouns, including complex names, news language generally comparatively large difference with the actual life, so the identification results.

Aishell use same accuracy is low, voice and data is too concentrated in specific areas, lack of life language voice and data, therefore easier to a phoneme in training and rare combination of synthesis for words at ordinary times, and can't identify the situation of the words it produces words generated situation, lead to the resulting low voice and data accuracy.

The free-ST model has a good effect, but it has high requirements on the pronunciation of words, and the accuracy rate is about 90%. It supports the recognition of some proper nouns and has a large reserve of words. It is the training set and training model that we finally use. We record the learning rate and the loss while training free-ST model.





It can be seen that the epoch value of the model converges rapidly within the range of 0-50, and the loss value continues to decrease. Training and convergence effect are good.

4.2 Speech emotion and gender recognition

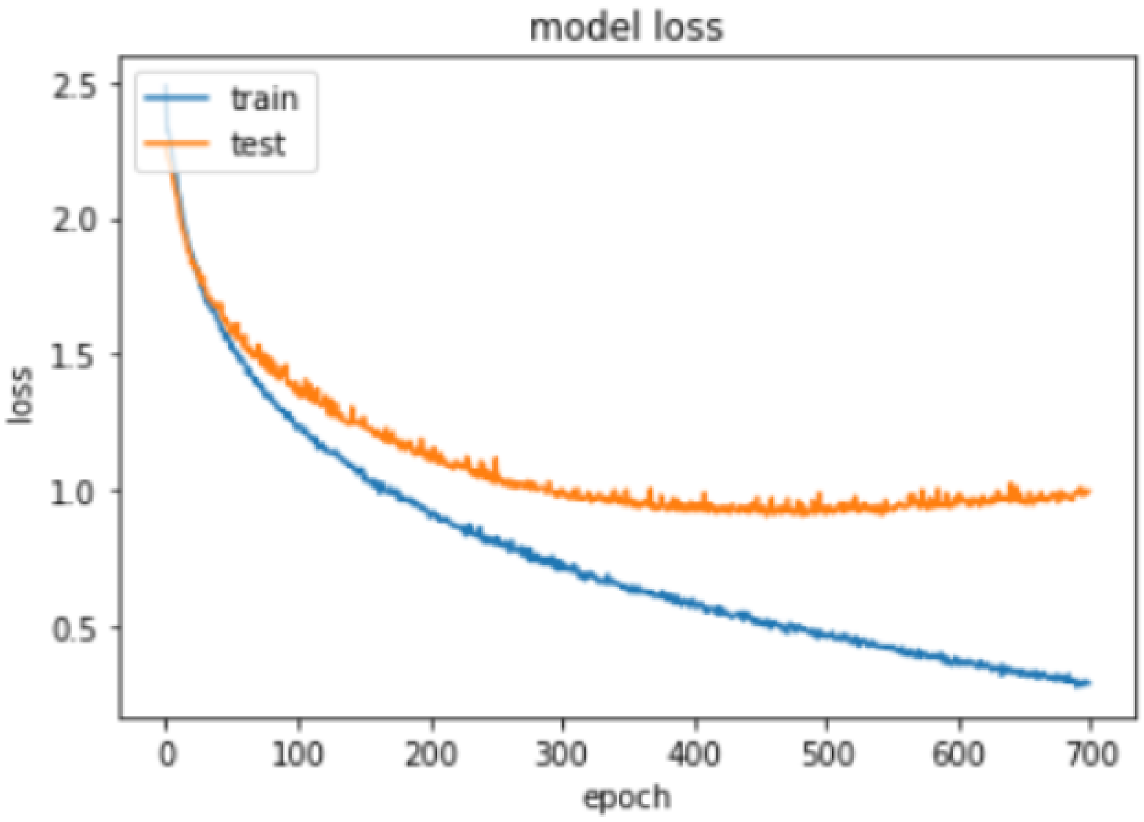
We evaluate the model saved after training, and we choose a test set that is completely different from the training set. The test set contains a total of 959 audio files. Similarly, we first use extract.py to extract the <path/label> list, which is the real label of the audio file; at the same time, we store the audio file prediction result (label) in Another list. By processing and comparing these two lists in multiple dimensions, the above evaluation parameters are calculated.

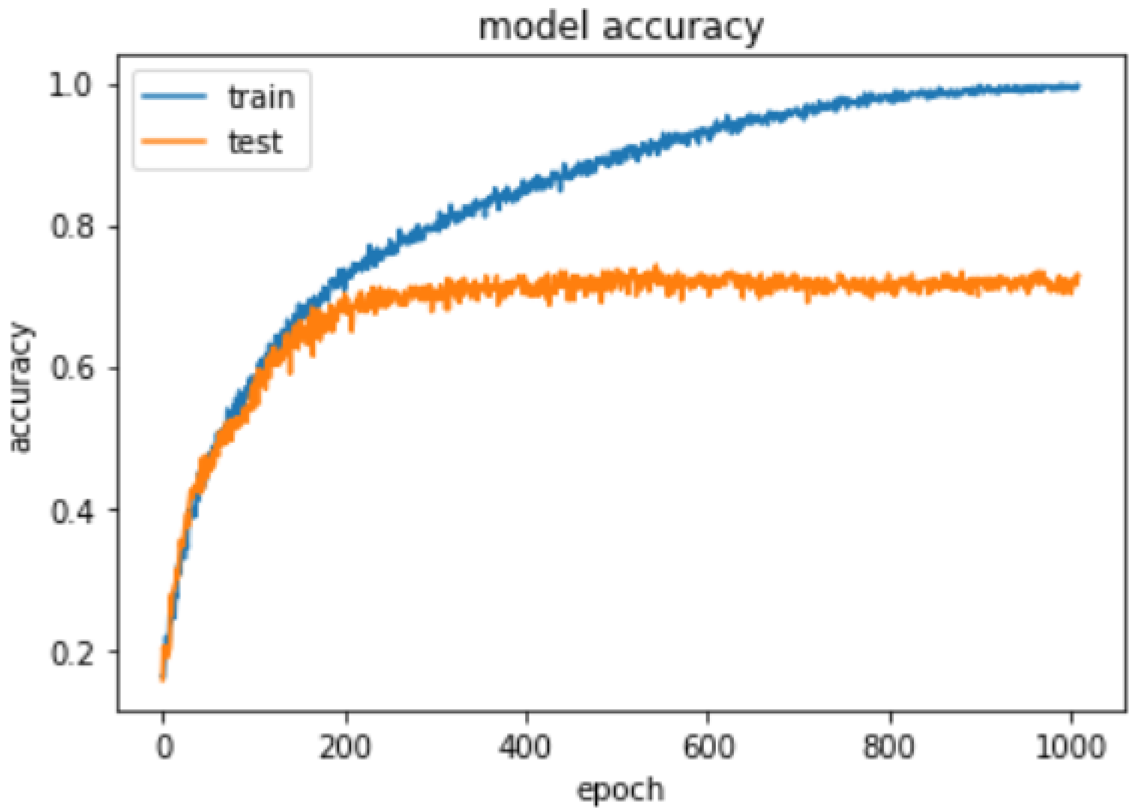
```
Micro precision 0.9103232533889468
Micro recall 0.9103232533889468
Micro f1-score 0.9103232533889468

Each class ROC
0.9892936461954422
0.9964271919660099
0.9850086906141368
0.9914662997296253
0.9901264967168791
0.9944838740826574
0.9922392787524366
0.9900661452298185
0.9872175550405563
0.9778872151409811

AUC: 0.989140679092956
```

It can be seen that the accuracy and regression rate of the model on the test set are very good, and the trained model is relatively good. We record the accuracy and the loss while training.





The training process and the loss and accuracy obtained after each round in the model training process are shown in the figure below. It can be seen that the model converges significantly faster after 200 epochs.

4.3 Voice print recognition and comparison

As mentioned above, the model training will be performed after each round of training, and the prediction model will be saved after the training. We use the prediction model to predict the audio features in the test set. Since the function of this module is to compare between models, audio features should be used to compare in pairs. The threshold value is from 0 to 1, and the step size is 0.01 for control, so as to find the best threshold value and calculate the accuracy rate.

```
-----------  Configuration Arguments -----------
input_shape: (1, 257, 257)
model_path: models/voice_model.h5
------------------------------------------------
提取特征...
100%|██████████████████████████████████████████████████|
5332/5332 [00:59<00:00, 57.28it/s]
对比特征...
100%|██████████████████████████████████████████████████|
5332/5332 [02:01<00:00, 41.68it/s]
100%|██████████████████████████████████████████████████| 100/100
[00:02<00:00, 31.70it/s]
当阈值为0.990000，准确率最大，准确率为：0.999693
```

It can be seen that the accuracy of the model on the test set is very good. Other parameters, such as regression rate, are not calculated because voiceprint comparison cannot provide standardized result information like multi-classification prediction.

5.Advantages & Disadvantages & TODOs

Above, the three main models constructed by us are described in detail, and the training results of the model are briefly analyzed. This part will summarize the three-part model constructed by the project for the three problems, and elaborate the advantages and disadvantages of the model as well as some TODOS of the project.

5.1 Chinese speech recognition

The speed of text transfer is fast, and the results can be obtained quickly after the audio is imported. The trained model can be imported quickly, and the model can be switched by changing a few simple file paths, which is convenient for users to use and manage. The predictive training process has strong controllability, and code generation and assembly is carried out through configuration, so that users do not need to find a needle in a haystack in the code, but meet the requirements of the modified model generation and use through the modification of config.

Turn words cannot reach the correct level of commercial, correct remains to be improved, the lack of noise reduction operation, you can use the related algorithm for noise reduction, voice input to support voice transfer text in noisy environment, higher requirements for input audio, does not support dialect, these are all after we can start to research and perfect direction.

5.2 Speech emotion and gender recognition

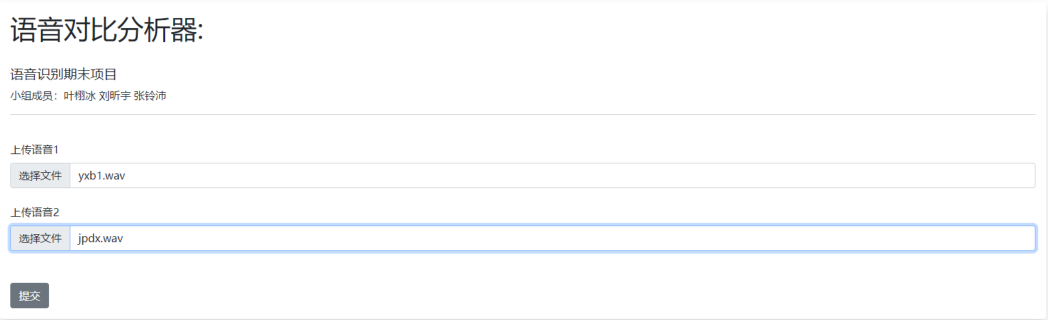
This part is an excellent realization of voice emotion prediction and gender prediction, and the predicted probability is reflected through the visualized radar chart. At the same time, the one-dimensional convolutional neural network model obtained by training has good accuracy and regression rate on a large number of test sets, indicating that the model has a high prediction success rate and can effectively predict individual audio, which achieves our original goal .

At the same time, this project also has certain defects. The first is that the length of the input audio is limited. Since emotion recognition requires that the input speech cannot be too short, the feature vectors are intercepted to 216 dimensions during model training. When the input audio features are less than 216 dimensions, prediction will not be possible . In addition, when the noise is large or the number of speakers is too large, the emotion prediction will be biased, because the voice may contain a variety of emotional characteristics and gender characteristics.

5.3 Voice print recognition and comparison

In this project, the model architecture was well designed in the contrast model of voice print, and the feature matrix generated by the residual neural network model obtained from training was used instead of the prediction result. In this way, the problem that the speech outside the test set could not be predicted was well solved and the original design goal was realized. This project still has some common shortcomings of voice print recognition applications. For example, the voice of the same person is changeable and easily affected by physical condition, age, emotion, etc. Different microphones and channels affect the recognition performance. Environmental noise interferes with identification; For example, in the case of mixed speaker, it is difficult to extract the characteristics of human voice print. However, in essence, SpenAugment we use in Chinese speech recognition has solved the problems of data voice deformation and noise very well. Therefore, the project will share the methods between models for the above problems in the future and solve the common problems.

6 Preview of interactive web



语音对比分析器:

语音识别期末项目

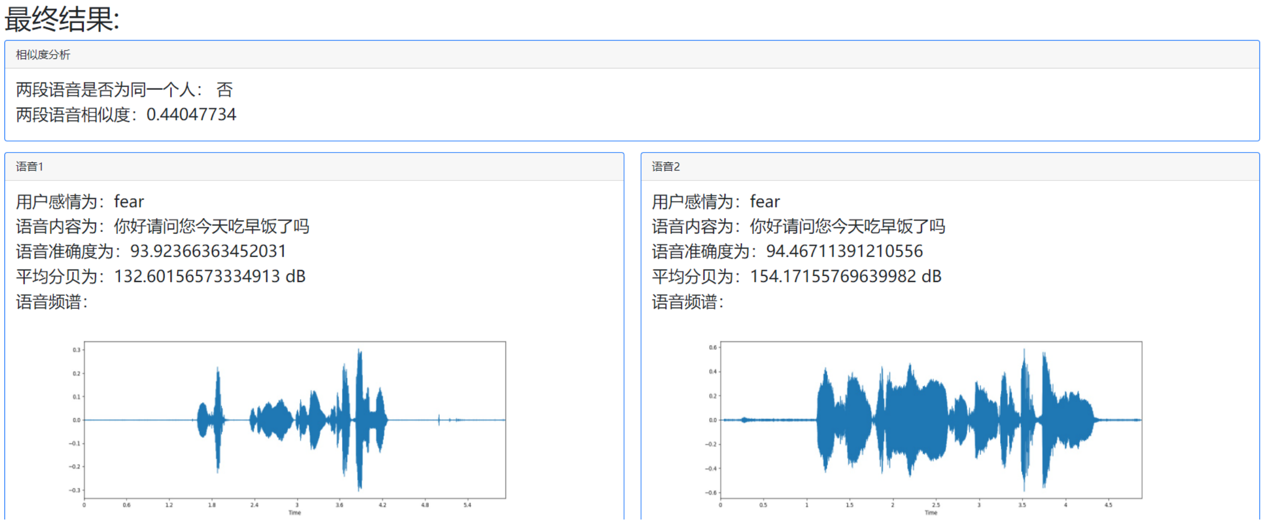
小组成员：叶栩冰 刘昕宇 张铃沛

上传语音1

yxb1.wav

上传语音2

jpdx.wav



最终结果:

相似度分析

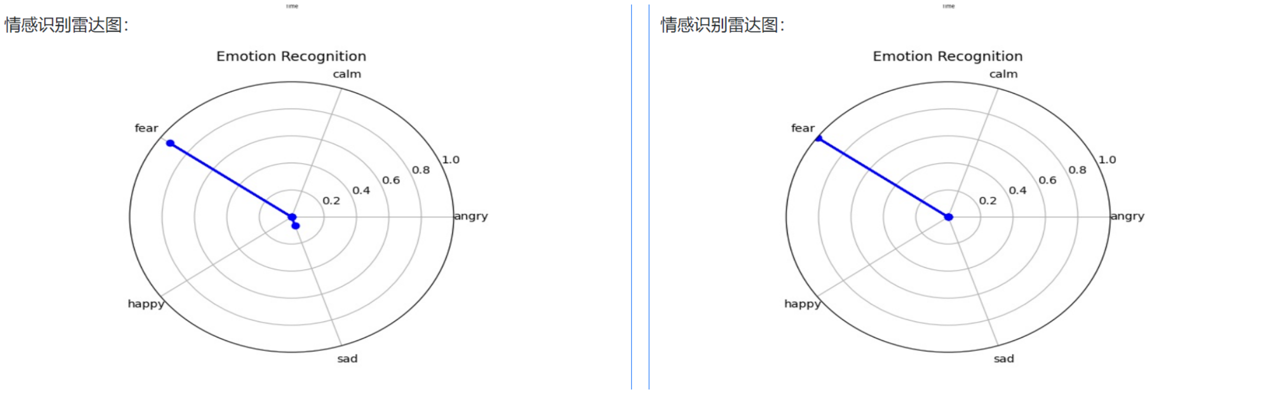
两段语音是否为同一个人： 否
两段语音相似度：0.44047734

语音1

用户感情为：fear
语音内容为：你好请问您今天吃早饭了吗
语音准确度为：93.92366363452031
平均分贝为：132.60156573334913 dB
语音频谱：

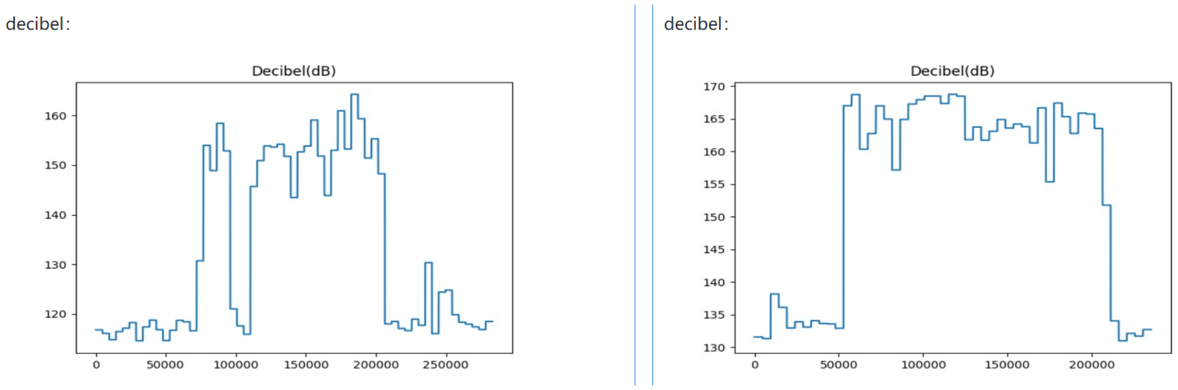
语音2

用户感情为：fear
语音内容为：你好请问您今天吃早饭了吗
语音准确度为：94.46711391210556
平均分贝为：154.17155769639982 dB
语音频谱：



情感识别雷达图：

情感识别雷达图：



decibel：

decibel：

All above is our project report, I wish you a smooth work, pleasant mood!