Smart Songs Selector: Find Your Voice

Bin Yu
Jiang Miao
Fu Zhan

Department of Electrical and Computer Engineering
Worcester Polytechnic Institute
May 3, 2016

Smart Songs Selector: Find Your Voice

Bin Yu
WPI
100 Institute Rd,
Worcester, MA 01609
byu2@wpi.edu

Jiang Miao
WPI
100 Institute Rd,
Worcester, MA 01609
jmiao@ wpi.edu

Fu Zhan
WPI
100 Institute Rd,
Worcester, MA 01609
fzhan@ wpi.edu

ABSTRACT
The smartphone market has been growing at a phenomenal rate. There is an increasing number of users using smartphone apps listening music or find music. Most People are in favor of singing and they hope to sing beautiful songs by imitating famous singers. Many websites provide music search services by recording voice when users sing, hum, or whistle. But most of these services are provided by web community and only aiming at search specific songs. They cannot find suitable songs for different kind of users. In this paper, we plane to design an android app which provides singer and song suggestions by comparing user voice feathers with famous singers'. We make two primary contributions. First, we develop a music app which can get the user voice file and communicate with server. Second, through analyzing feathers of users' voice via MFCC collected on many devices, we improve algorithms and make it much more accurate to analyze and recommend songs.

Categories and Subject Descriptors
H.1.2 User/Machine Systems; I.5 Pattern Recognition; J.3 Life and Medical Science

General Terms
Algorithms, Management, Measurement, Experimentation

Keywords
MFCC; data analysis; behavioral trends

1. INTRODUCTION
The Smartphone App industry is in the growth stage of its life cycle. In the 10 years to 2020, industry value added, a measure of the contribution of the industry to the overall economy, is expected to grow at an annualized 37.0%.[1] Despite the phenomenal market penetration of smartphone apps, there is few music apps provide songs recommendation service. For those who are willing to practice singing, an unsuitable song may leads to a feeling of frustration and thus results in a failure of study music. In this paper, we develop an app whose name is "Find your voice" and we focus on the voice of user. Most people are in favor of singing and they hope to have beautiful songs to get attention from others. But they might choose wrong songs which are difficult to perform. Hence the way to get the right voice guidance and choosing songs is very important. As we all know, each person's voice is unique, but you can still find the similar voice of singers whose songs may be suitable for you. We aim to design an android app which provides singer and song recommendations by analyzing voice feathers. Users can clearly know which singers are suitable for themselves and they can choose songs from those to practice.

2. RELATED WORK

2.1 About the app market
In the developing world, one of the most common ways to find a song is via song names or lyrics. There is an increasing number of music searching engine by sound because many people only know the tune to a song. There are a ton of ways to identify songs easily with online tools, desktop apps and smartphone apps such as Midomi, AudioTag, SoundHound and so on. The smartphone's instant popularity and high adoption rate stimulated developers to quickly offering entertainment, lifestyle apps especially in music area. In the coming years. Developers are expected to increasingly rely on the development of mobile music apps that are integrated with smartphone operating systems and less on the development of web apps. What's more, most apps in smartphone and web are not support recommend songs according to users' voice but only rely on users' listening hobby. Above two aspects herald a huge potential market.

2.2 About MFCC Algorithm
Extracting the feature parameters which are characterizing the speaker is the most important part. The currently most widely used MFCC parameter primarily describes the spectrum envelope of the sound tract characteristics and ignores the impacts of fundamental frequency theoretically.[2] But it is not perfect because the fundamental frequency may influence the description accuracy of MFCC parameters about the sound track characteristics. There is good way to improve MFCC algorithms which is smoothing MFCC, which is based on smoothing short-term spectral amplitude envelope. Smoking MFCC parameters can degrade the bad influences of fundamental frequency effectively and upgrade the performances of speaker recognition.

3. METHODLOGY
To extract the feather of human voice, the best method will be MFCC which is Mel-frequency cestrum coefficients. Basically, for a voice data, we first do framing and then transmit them into frequency domain (FFT). Mel filtering will be made to transmit data from regular frequency domain into Mel-frequency. Then we take the logarithm and do DCT (Discrete cosine transform), delta MFCC can be figured out by this method.[3] Vector quantization is necessary to generate the codebook. Once we have enough MFCCs of singers, it is possible to obtain good codebooks vie

LBG algorithm which is able to determine your voice is similar to whom.

4. IMPLEMENTION

We want to build this project from two different parts: App (Mobile), Server (Computer), the framework is shown as the following Figure 1:

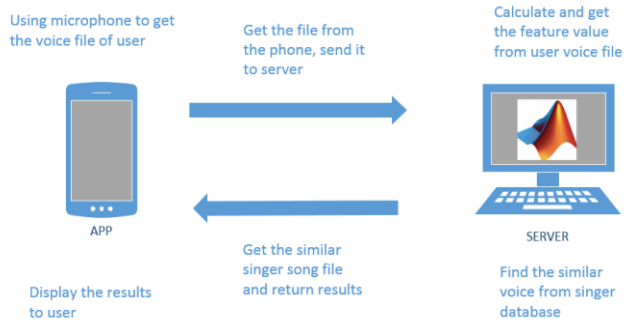
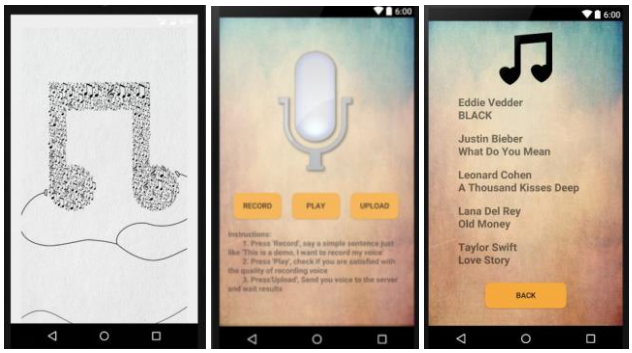


Figure 1. Whole Framework of our project

The mobile part is responsible for get the users' voice recording files and transfer them to our background processing part—'The Server'. The server is combined by two parts: one is written by Java to finish the file transfer function. Other part is designed by Matlab to complete the voice recognition and match function. We use Eclipse to write the sever code to call the Matlab function so that we can combine these two parts together.

4.1 App on the mobile

Before we design the UI of our app, we think the key point for user experience of an App are simple operation and response fast. So we hope our app that doesn't have some not practical functions, use only need to use few simple buttons to get what they want. Our interface is clear and fast, it shown on Figure 2:



a) Splash b) Record c) Result

Figure 2. Three different screenshots of our App

When user open this app, at first they will see the splash window to tell them our App theme is music. After about three seconds, it will jump to the Record window. User can press and hold the Record button, use mobile microphone to say a simple sentence or sing a song. After finish recording process, users can press the Play button to listen they just recorded voice. If they are satisfied with the quality of voice, they can press Upload button to send this file to the server part. After few seconds, the result window will display the singers name and their songs which we think the voice of these singers is similar to your voice, you can choose these songs to practice.

4.2 Server on the computer

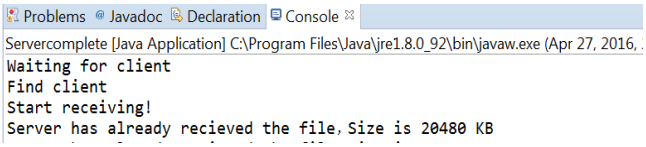
Though this part we finish the voice phrase processing, generate the result and send it to the App

4.2.1 How to transfer the file

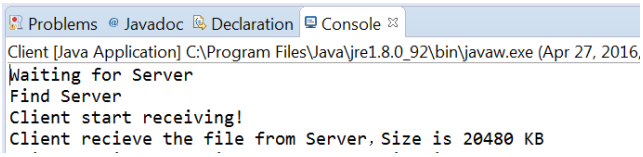
To ensure the stability of connection between the App and server, we plan to use TCP protocol to finish the transferring process. Java also afford this class which name is 'ServerSocket'.

Sockets provide the communication mechanism between two computers using TCP. A client program creates a socket on its end of the communication and attempts to connect that socket to a server.[4] When the connection is made, the server creates a socket object on its end of the communication. The client and server can now communicate by writing to and reading from the socket.

So what we need to do is just get the IP address and port number. The connection is safety and fast. Figure 3 shows the test results of transferring process:



a) App upload the file to the Server



b) Server return the result file to the App

Figure 3. Testing result of transferring process

4.2.2 How to get the feature of the voice

The Mel scale relates perceived frequency, or pitch, of a pure tone to its actual measured frequency. Humans are much better at discerning small changes in pitch at low frequencies than they are at high frequencies.

The formula for converting from frequency to Mel scale is:

$$M(f) = 1125 \ln(1 + f/700) \quad (1)$$

To go from Mels back to frequency:

$$M^{-1}(m) = 700(\exp(m/1125) - 1) \quad (2)$$

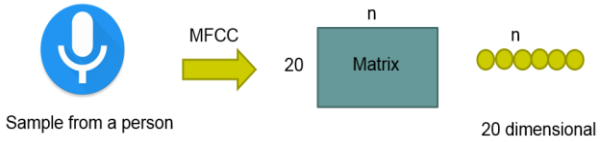


Figure 4. MFCC Algorithm

For a sample as shown in Figure 4, we first frame the signal into small pieces (100 points for each piece with overlapping) and perform DCT for each frame. In this way, we are able to get the 20*n matrix. 20 MFCC coefficients are extracted for each frame and n depends on the sample length.

To generate the codebook, Vector Quantization is performed. It is a classical quantization technique from signal processing that allows the modeling of probability density functions by the distribution of prototype vectors.[5] Basically, we use LBG algorithm to generate 256 codewords. For each sample points to its closest codeword, the sum of those distance are calculated to be the smallest. As shown in Figure.5, the codebook is a 20*256 matrix.

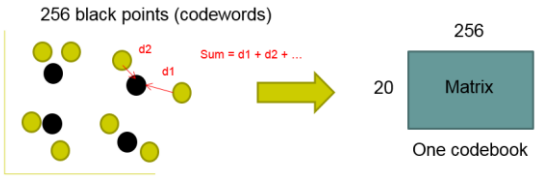


Figure 5. Vector Quantization

For user sample, as shown in Figure 6, we calculate the total distance to codewords and compare the result of all codebooks to figure out the smallest distance which means they should be the similar voice.

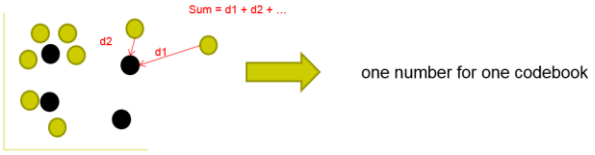


Figure 6. Calculate distance to codebook

4.2.3 How to complete connection between App and Server

To complete this target, we use Java Builder JA toolbox to deploy Matlab script/function as java class, then add the (converted) java class in to your java project library, and additionally the Java builder jar file. Through this method we combine this two tools together and generate a series of data to judge our experiment results.[6]

Hence we use the Matlab to generate two JAR files of MFCC algorithm, make sure it can phrase the voice file from the App. Figure 7 shown all the running process of our server, finally it will generate a result file which contains match degree of different singer and return it to the App.

```
ServerReceive [Java Application] C:\Program Files\Java\jre1.8.0_92\bin\javaw.exe (May 3, 2016, 9:49:29 PM)
Server: Waiting for Client!
Finding Upload File!
************************************************
The same file alread exists on this path, begin overwriting!
Server has already received the file, the size is 20480 KB
Server has already received the file, the size is 10416 KB
Server has already received the file, the size is 3472 KB
Server has already received the file, the size is 3472 KB
```

a) Detect the upload file from app

```
ServerReceive [Java Application] C:\Program Files\Java\jre1.8.0_92\bin\javaw.exe (May 3, 2016, 9:49:29 PM)
Server has already received the file, the size is 352 KB
Server has already received the file, the size is 3472 KB
Server has already received the file, the size is 13888 KB
Server has already received the file, the size is 1628 KB
Server Receiving Success! Waiting for generate result file!
Generate the phrasing result! Start sending it to App!
************************************************
Server send the result file to App, the size is 146 KB
Server has already finished the sending process
```

b) Generate the new file to APP

Figure 7. Console result of whole server

Through this method, we make Java and Matlab working together and implement the two-way transmission between App and server

5. Evaluation

When we get the feature value of the voice, then we can use it as a standard to judge if two voices are similar or not. So we can do a series of experiment to prove our algorithm is accurate and efficiency.

5.1.1 Same kind voice detection

At first, we choose 5 students to record a same simple sentence "This is a demo". We will use these five voice files as the codebook. Then we ask these students to say the same sentence again and record them as the test target to describe the quality of the match with the codebook. The result is shown as the Table 1.

	A	B	C	D	E
cbA demo	4.17E+03	5.95E+03	5.34E+03	4.98E+03	6.35E+03
cbB demo	5.94E+03	5.07E+03	5.73E+03	5.68E+03	6.27E+03
cbC demo	5.14E+03	6.36E+03	5.08E+03	5.55E+03	6.87E+03
cbD demo	5.13E+03	6.14E+03	5.97E+03	4.68E+03	6.55E+03
cbE demo	6.73E+03	6.99E+03	7.75E+03	6.39E+03	3.97E+03

Table 1. Same 5 students saying same sentences

Second, we ask each student to say a different sentence and recording them as the testing sample, through Matlab to calculate the feature value. Compare with the codebook, the result is shown as the Table 2

	A	B	C	D	E
cbA demo	5.66E+03	5.95E+03	7.08E+03	5.78E+03	8.92E+03
cbB demo	6.31E+03	5.46E+03	7.08E+03	6.14E+03	7.78E+03
cbC demo	5.81E+03	6.04E+03	7.08E+03	6.27E+03	9.43E+03
cbD demo	6.47E+03	6.18E+03	7.73E+03	6.16E+03	8.78E+03
cbE demo	7.05E+03	6.64E+03	7.92E+03	6.16E+03	7.26E+03

Table 2. Same 5 students saying different sentences

At last, we choose five different singers talking show and cut out a part as the codebook, then cut out another part as the testing sample, the result is shown as the Table 3

	Neyo - part2	Mars - part2	Adele - part2	Taylor - part2	Beyonce - part2
Neyo - part1	9.31E+03	1.18E+04	6.74E+03	7.19E+03	1.56E+04
Mars - part1	1.24E+04	1.11E+04	9.08E+03	8.22E+03	2.05E+04
Adele - part1	1.25E+04	1.42E+04	6.84E+03	6.19E+03	1.29E+04
Taylor - part1	1.29E+04	1.45E+04	7.52E+03	5.31E+03	1.30E+04
Beyonce - part1	1.29E+04	1.54E+04	6.92E+03	6.50E+03	1.04E+04

Table 3. 5 different singers talking part

We know that the distance between codebook and testing sample becomes smaller, their degree of match becomes higher. Hence through analysis the data we can clearly find only one group (The blue line) is not match, the rest parts (The red line) are all match their own voice. We think may be the voice of Student D is similar with Student A, Ne-Yo's tone are lower and deep, same with Adele, and also their voice pitch are on the same line of sound which may cause this result happened. Overall, the test result is accurate, we think our algorithm can easily distinguish if

the voice is came from the same people. It can detect the voice belongs to whom.

5.1.2 Different kind voice detection

In this part, we choose five different singers whose name are: Ne-Yo, Bruno Mars, Adele, Taylor Swift, Barbra Streisand, Justin Bieber and Linkin Park. We choose a song from each of them as the codebook, then we choose 4 songs from different people which their gender and voice are obviously different. Calculate the distance and the result is shown as the Table 4

	A, male, high	B, female, low	C, male, high	D, female, low
So Sick, Ne-yo	2.98E+04	4.26E+04	4.34E+04	6.98E+04
Grenade, Bruno Mars	2.62E+04	4.53E+04	3.94E+04	6.79E+04
Rolling In The Deep, Adele	3.45E+04	6.05E+04	4.74E+04	9.30E+04
Love Story, Taylor Swift	2.99E+04	5.40E+04	4.12E+04	6.28E+04
The Windmills Of Your Mind, Barbra	2.81E+04	4.66E+04	4.14E+04	5.50E+04
What Do You Mean, Justin Bieber	2.64E+04	4.19E+04	4.74E+04	6.66E+04
Numb, Linkin Park	3.29E+04	6.25E+04	4.73E+04	9.16E+04

Table 4. 4 different people's song comparing with the singer

The user A is a young male and he sang a fast song which its rhythm is strong. The user B is a young female and her song's pitch is lower but rhythm is also fast and crisp. The user C is a mid-age male and he sang a rock song. The last user D is an old female and her pitch is very lower and deep. The test result prove their voice condition. Bruno Mars's voice is high and sharp. Both of these two males sang passion songs which make their voice one the higher pitch. For women, Justin Bieber's voice is not very high and can be easily reached. Hence B sang a song with normal pitch may adapt this tone level. The D's voice is the most special, because her voice is typically lower and husky. So it has no doubt be match with Barbara because her voice is the deepest in all these singers. Our algorithm can react to the voice which their pitch are obviously different.

6. DISSCUSSION

In this section, we describe the technical challenges and how to improve the accuracy of voice recognition. We find some useful method to help computer to get the right choice of voice, not through our ears, but through the algorithm.

6.1 Technical challenges

During the field study, we faced three major technical issues.

The first one is how to get the high quality songs of different singers. The most important thing in voice match is get the codebook from test sample and matching object. For test sample, the recording environment is very critical, we'd better to find the place as quiet as possible before recording the voice. The other problem is how to reduce the noise of the voice file. In our project we use Audacity to finish this processing, but the quality of voice file is still not very good. We have to repeat this process so many times to get the high quality file which wastes a lot of time. For matching object, which means the songs of singer. What we need is to get the pure vocal version which name is 'Acapella'. This version is very efficiency because it has no background music. We can easily get the codebook from it. However, it is finished in the studio with singer alone and very hard to find on the market, especially to some old and classical songs. Hence find the higher quality source voice file is the precondition of all the work in our project.

Second, through a lot of testing work in our project, the accuracy of match degree is not satisfactory. Although we assume other external factors such as noise, length of recording file cannot influence the result, there are still many attributes of voice and effect the result. Such as pitch of voice, rhythm of songs, dynamics of speaking power, tonality of melody and so on. Hence we cannot build a specific standard to measure the match degree. In our project we only focus on the pitch of voice and despite other factors. But if the user want to get much higher accuracy. We need to build a multidimensional voice model which includes all these factors.

At last, how to make the app and sever work together is the core problem in our design. Our project need to combine Android, Java and Matlab. Hence how to connect each part is very import. In the process of testing, App connect with server is easy but handling the received file is hard, in addition we need this process must be finished as soon as possible. Because user's patience is limited and they can't wait the process for the long time. Especially this issue occur in Matlab. Hence improve the calculating speed of server part is necessary.

6.2 Improve the voice recognition accuracy

For the single user, the easiest way to improve accuracy is speaking slower and continuous, choose a quiet environment to record your voice, sing clearly and ensure your voice will not waver or crack.

For the calculation process, although MFCC is the most widely used feature in voice recognition, it is very sensitive to noise interference, which leads to drastically degrade the performance of recognition systems because of the mismatches occur in testing process. MFCC feature analysis is very successful except when the noise is present.[7] Hence what we can do is use one-sided autocorrelation sequences of speech instead of original speech, because autocorrelation of the noise in many cases could be considered relatively constant over time so a high pass filtering could lead to suppress the noise furthermore.

7. CONCLUSION AND FUTURE WORK

In this paper, we presented the potential market in the songs recommendation app based on voice analysis and the whole process of app developing. Firstly according to the key result of a most popular app survey, we found a potential area in music apps. Thus we then developed a mobile music app. The app served as a tool for users to analyze their voice and then get some recommendation songs from famous singers whose voice is similar to the specific user. We had tested our application by male and female, also analyzed the testing results. We also improve the MFCC algorithms to get better results. In general there may be some differences of voice feature value between male and female. However, there are some user's voice are very close. If the feather value is close to many singers' then the app will provide several most closet singers' songs to the user.

As for the future work, we will build a more comprehensive database of searchable music. Because limited for the testing sample, our app can't match all the possible kinds' voice, we hope that our App users can contribute to build the database by singing in app online recording studio in any language or genre in the future. These voice will be stored in a different database from famous singers' voice database. The next time anyone searches for similar voice, someone's previous performance might be the result. This means users can create his or her own profile, sing favorite songs and get discovered by other users. Other side is develop the algorithm further, especially on noise-reduction field, which is the major influence to voice recognition. Future work

should also involve optimizing multithread processing on server part. We hope it can achieve good performance when many people testing their voice at the same time.

8. REFERENCES

[1] Kondrak, Grzegorz, and Tarek Sherif. "Evaluation of several phonetic similarity algorithms on the task of cognate identification." Proceedings of the Workshop on Linguistic Distances. Association for Computational Linguistics, 2006.

[2] Shi, Yibo, and Li Wang. "Improved MFCC algorithm in speaker recognition system." 2011 International Conference on Graphic and Image Processing. International Society for Optics and Photonics, 2011.

[3] Hong Lu, Wei Pan, Nicholas D. Lane, Tanzeem Choudhury and Andrew T. Campbell, SoundSense: Scalable Sound Sensing for People-Centric Applications on Mobile Phones

[4] Abhijit A. Sawant, Dr. B. B. Meshram, 2013. Network programming in Java using Socket, 2248-9622. DOI= 10.1.1.448.9867&rep=rep1&type=pdf

[5] M. McKinney and J. Breebaart. Features for audio and music classification. In Proc. ISMIR, pages 151.158, 2003.

[6] A. Naderlinger, Josef Templ, S. Resmerita, W. Pree, 2011, An Asynchronous Java Interface to MATLAB

[7] WU Zunjing, CAO Zhigang, 2005, Improved MFCC-Based Feature for Robust Speaker Identification, 1007-0214