Bird Species Recognition Using

Audio Data

THE UNIVERSITY OF

MELBOURNE

Yun Wang
672323
Supervisor: Richard Sinnott
Software Development Project
COMP90019 Distributed Computing Project (25 points)

Table of Contents

Abstract

We built a software to recognize species of birds using audio data automatically. The Mel Frequency Cepstral Coefficient is used for feature extraction, and Gaussian Mixture Model is used as the machine learning algorithm. In addition, we also used the MongoDB database to store the data and designed a web page to visualize the prediction.

Keywords: species recognition, machine learning, Mel Frequency Cepstral Coefficient, Gaussian Mixture Model

Acknowledgement

Firstly, I would like to express my gratitude to my supervisor Richard Sinnott for his great support and guidance during the process of accomplishing this project.

Secondly, I would like to thank my boyfriend for his company during my study.

Last but not least, I would like to thank my parents and grandparents for their financial support and love.

Statement of own work

I certify that this project is my own work, based on my personal research and study. All the material involved has been acknowledged.

1. Background

Collecting biodiversity data, such as the count of birds in a region, has always required human participation, which is not only a very limited resource compared to the scale of nature, but also has a high cost. Recent developments in computer science and artificial intelligence makes the automation of this procedure possible.

Many researchers have investigated the topic of recognizing bird species using acoustic data, i.e. the recordings of bird call. Most of them have access to large databases from scientific organizations, and some even collected the data by diving deep into the countryside to do the recording themselves [1].

Feature extraction and machine learning algorithms are the most crucial parts of species recognition. Miguel, Carlos and Hector proposed a feature extraction method [1], which uses a vector of length 4 to represent the sound structure for 9 frogs and 3 birds, including minimum frequency, maximum frequency, maximum power and call duration. Although the feature space seems simple, the prediction accuracy is actually high thanks to their extensive training data—more than 10,000 recordings of birds and frogs. Anuradha and Upul used a feed-forward artificial neural network to categorize a selected set of 6 Sri Lankan bird species, which also achieved satisfying results [2]. However, the configuration of neural network is complicated and is unlikely to get good results in a very limited amount of time.

Davis and Mermelstein proposed a feature extraction algorithm for monosyllabic word recognition called Mel-frequency cepstral coefficients (MFCC) which soon became state of art [3]. It is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency [4]. Other researchers also investigated the use of MFCC in different fields such as musical instrument identification [5].

2. Introduction

In our project, bird sounds for both training and testing is downloaded from xeno-canto(http://www.xeno-canto.org/), which is a website dedicated to sharing bird sounds from all over the world, however in this project we are only concerned about bird species identification in Australia.

Due to the limited amount of training data (around 30 recordings for each bird species), Mel-frequency cepstral coefficients is chosen as the feature extraction method. MFCC allows us divide a single recording (several minutes long) into hundreds of frames and calculate a feature vector for each frame [4]. Since bird calls are constantly repeating during the period of the recording, representing the whole recording using only one feature vector would omit some important information. Using MFCC enables us to expand from one feature vector to thousands of feature vectors for each recording, thus capturing the repeated pattern. MFCC is well-known for its excellent performance in human speech recognition, while in this project, we made a novel experiment by applying it to the recognition of bird sounds.

Gaussian mixture model is widely used to approximate the distribution of features in a biometric system, such as vocal-tract related spectral features in a speaker recognition system. Its capability of representing a large class of sample distributions turns out to be also helpful in bird species recognition, since they share many common traits with human speech

[6]. However, instead of applying Gaussian mixture model in a traditional way, where only one Gaussian mixture model is trained for all the classes, we used a better approach through training a model for each bird species, and during testing, return the bird species whose model gives the highest likelihood for a recording.

In addition to feature extraction and machine learning, our project also includes a front-end webpage and a back-end database, which allows users to upload recordings of bird calls (audio files). Calculations are performed on species prediction and the confidence level is returned and shown on the webpage. Since the geographical information is also included in the name of the file the user uploaded, location information will also be visualized. To improve the accuracy of prediction, the size of database will grow over time, by storing data with confident predictions.

3. Data collection

In order to build a machine learning model, we need to first collect bird sound data for training. There are not many bird sound resources accessible to the public, some of the databases require users to pay depending on the size of the data they need. Others belong to private scientific organizations or are collected by the researchers themselves. Due to a lack of devices and human resources, recording the sound on our own is not realistic. Fortunately, the xeno-canto website has recordings of wild birds from all across the world. The website allows users to see the statistics of recordings files based on regions. For our project, the bird species that is recorded the most in Australasia is shown in Figure 1. The audio files are mostly recorded by volunteers, i.e. bird lovers, tour guides, biologists, zoologists and so on. Due to the diversity of recordists and devices, the recordings vary in quality, sample rate and duration.

Figure 1. Species most frequently recorded in the Australasia collection with the number of recordings (The ranking has slightly changed since the files were downloaded).

To collect data, we make use of the xeno-canto API [7] to download audio files automatically. The nine bird species downloaded are Australian Golden Whistler, Australian Magpie, Brush Cuckoo, Eastern Whipbird, Grey Fantail, Grey Shrikethrush, Pied Currawong,

Southern Boobook, Spangled Drongo. Only “A” quality (the best quality labeled by xeno-canto) files are downloaded, in order to avoid lower quality recordings misleading the classifiers, as most of the recordings are uploaded by amateurs and qualities are not guaranteed.

The name and quality of recordings can be specified in the query. The xeno-canto API returns JSON objects with fields as shown in Figure 2 and Figure 3. As we can see, there is a URL which can be used for downloading recordings.


```
{
    "id":"134880",
    "gen":"Pheucticus",
    "sp":"ludovicianus",
    "ssp":"",
    "en":"Rose-breasted Grosbeak",
    "rec":"Jonathon Jongsma",
    "cnt":"United States",
    "loc":"Grey Cloud Dunes SNA, Washington, Minnesota",
    "lat":"44.793",
    "lng":"-92.962",
    "type":"song",
    "file":"http:\/\/www.xeno-canto.org\/134880\/download",
    "lic":"http:\/\/creativecommons.org\/licenses\/by-sa\/3.0\/",
    "url":"http:\/\/www.xeno-canto.org\/134880",
    "q":"A",
    "time":"9:16",
    "date":"2013-05-25"
}
```


Figure 2. Example JSON object returned for a query.

- **id**: the catalogue number of the recording on xeno-canto
- **gen**: the generic name of the species
- **sp**: the specific name of the species
- **ssp**: the subspecies name
- **en**: the English name of the species
- **rec**: the name of the recordist
- **cnt**: the country where the recording was made
- **loc**: the name of the locality
- **lat**: the latitude of the recording in decimal coordinates
- **lng**: the longitude of the recording in decimal coordinates
- **type**: the sound type of the recording (e.g. 'call', 'song', etc). This is generally a comma-separated list of sound types.
- **file**: the URL to the audio file
- **lic**: the URL describing the license of this recording
- **url**: the URL specifying the details of this recording
- **q**: the current quality rating for the recording
- **time**: the time of day that the recording was made
- **date**: the date that the recording was made

Figure 3. Explanation of the fields in JSON object.

4. Data preprocessing

Data preprocessing can be divided into two parts, the first part is to transform the format and sample rate of audio files using the “iTunes” app, and the second part is to extract features.

4.1 Audio format transformation

Once that the audio files are ready, we need to consider transforming them to a uniform format so that they can be fed into our feature extraction algorithm. Audio waves are typically sampled at 44100 Hz and the feature extraction algorithm needs “wav” format as input. iTunes is used to transform the audio files in this project. In addition, all audio files are also labeled with names like “BrushCuckcoo_23.wav”, so that the names can be used as class

labels during training and testing. When users want to get prediction results from the website, they also need to include location information in file names.

4.2 Feature extraction algorithm Mel Frequency Cepstral Coefficient

In order to recognize the species of bird from the audio data, features first need to be extracted. Mel Frequency Cepstral Coefficient (MFCC) values are widely used in automatic speech and speaker recognition. It is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency [4]. Since the time this algorithm was introduced by Davis and Mermelstein in the 1980s, it has been state of the art.

Sounds generated by human are filtered by their vocal tract, consisting of tongue and teeth. If we could find out what the shape is, then the phoneme being produced can be represented accurately. The shape of the vocal tract is reflected by a short envelope of the power spectrum, which can be described by MFCCs precisely. MFCC also considers human perception sensitivity in terms of frequency. The relationship between perceived frequency (Mel scale) and actual measured frequency is shown as the following formula and in Figure 4:

$$M(f) = 1125 \ln\left(1 + \frac{f}{700}\right)$$

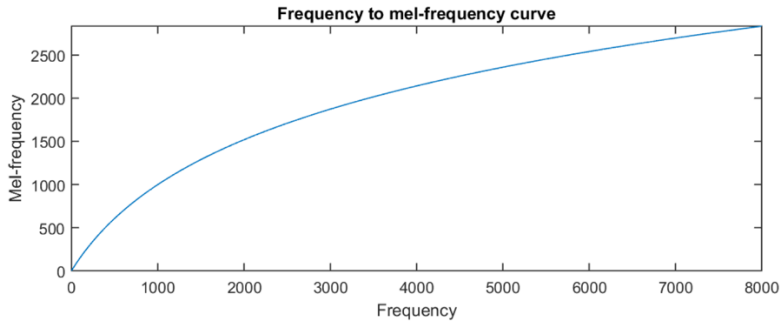

Figure 4. The relationship between actual frequency and perceived frequency (Mel frequency)

As we can see from Figure 5, the human ears do not differentiate high frequencies as well as low frequencies. Although MFCC is mostly applied to human voice, in this project, we made a reasonable assumption that there are adequate similarities between the human voice and bird sound, which is proved to be sufficient given our final prediction results.

The basic steps to calculate MFCCs are as follows [4]:

1). Frame the signal into short frames.

Audio signals are changing all the time, however, in a small time scale, we may assume that the audio signals are statistically stationary. Therefore, the acoustic data is sliced into frames which are 20-40 ms long. If the frame is too long, the audio data would change too much during this period; if it is too short, there are not enough samples to represent the spectral feature.

2). For each frame calculate the periodogram estimate of the power spectrum.

The periodogram can identify different frequencies presented in the sound, which resembles the human cochlea structure. Depending on the frequency of incoming sounds, different areas of human cochlea vibrate, which triggers different nerves to inform the brain about certain frequencies.

3). Apply the Mel filterbank to the power spectra, sum the energy in each filter.

The power spectra still contains some information that is not needed for automatic speech recognition, especially given that humans can not differentiate between two closely spaced high frequencies. To get a rough representation of how much energy exists in each frequency bin, we sum up the periodogram inside the frequency region. The width of each bin is chosen using the Mel Scale, which defines the spacing of each filterbank. The bin is very narrow near 0 Hz at first and as the frequency increases, the bin becomes wider since we are less concerned about the variations in the high frequency region. An example Mel-filterbank is shown in Figure 5.

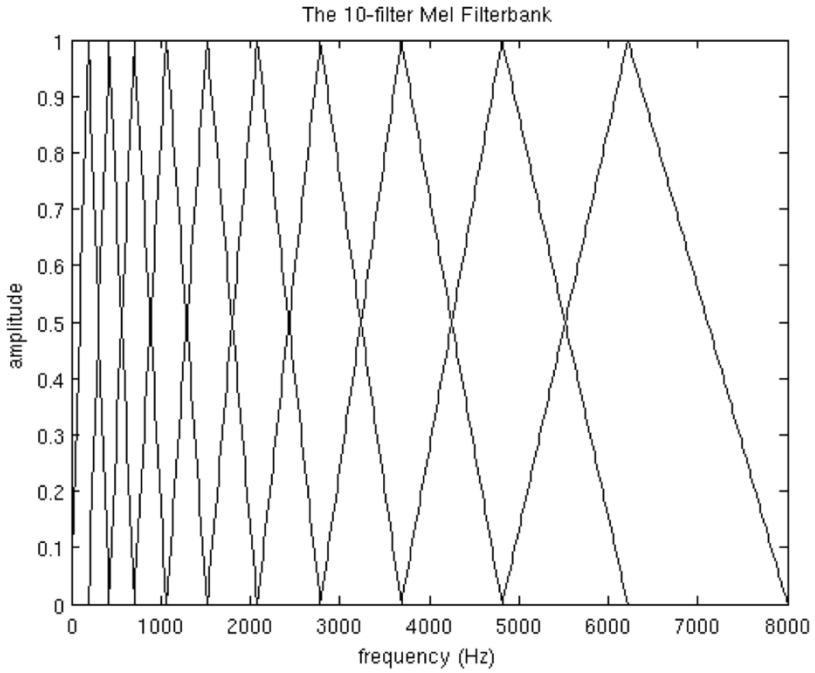

Figure 5. A Mel-filterbank containing 10 filters, starting at 0 Hz and ending at 8000Hz (The actual implementation may be slightly different).

4). Take the logarithm of all filterbank energies

This step is also inspired by human hearing: people do not hear loudness on a linear scale. In order to double the perceived loudness, 8 times as much energy is needed, which suggests that when energy increases, the volume people can hear does not actually change much. For this reason, the logarithm of energies is calculated to simulate human perception.

5). Take discrete cosine transform of the log filterbank energies.

The filter banks used in the previous step are actually overlapping with each other, and as a result the energies filtered are correlated. A discrete cosine transform (DCT) expresses a finite sequence of data points in terms of a sum of cosine functions oscillating at different frequencies, which helps to decorrelate the energies and allow features to be represented using diagonal covariance matrices.

6). Keep DCT coefficients 2-13, discard the rest.

After the previous steps, we are left with 26 cepstral coefficients, but generally only the first 12 ~13 coefficients are kept. This is due to the fact that the higher coefficients represent faster changes in energy variation and are actually detrimental to the automatic speech recognition. By discarding them, we are able to gain a small amount of performance improvement.

A major drawback of MFCC values is that they are not very robust to background noise, and a general measure to lessen the influence of this is to normalize the values before applying them to our bird species recognition system.

In our implementation of MFCC, 13 coefficients are kept for each audio frame, and the length of each frame is 1024 samples. The overlap between each frame is half the length of the frame, 512 samples, which means the first frame starts at sample 0, and the second frame starts at sample 512, so on and so forth. The feature array for an audio file has thousands of rows and 13 columns, where each row represents the extracted coefficients for a frame.

5. Data storage

After extracting the features, a back-end database is used to store and manage the features for future use. For our project, the data structure is simple— we only need to store the file name and its corresponding features, so MongoDB is selected.

MongoDB is a cross-platform document-oriented database [8], which is a NoSQL database suited for JSON-like documents with dynamic schemas. It is a free and open-source database management system most popular for document stores.

The JSON object stored in MongoDB has the format as shown in the following example:

{'_id': "backup/AustralianGoldenWhistler_25.wav", 'feature':[[
-0.12803098814034636,
-7.055248496863598,
-0.07536362188228085,
-4.534714896849498,
0.5251429964613644,
-2.6265554063414447,
1.5094177135973403,
-0.0904062511559402,
0.04837714097015211,
1.1457401799882152,
0.19021597175540228,
0.5277208349268985,
-1.1272767245932291
],[...],[...]] }

Here "_id" denotes the field plays the same role as primary key in a relational database, and is set to the relative path of the audio file to avoid storing a same audio file twice. "feature" is another attribute with the extracted MFCC values as its content, as shown above, which is an array of arrays, and each small array holds the 13 coefficients of the corresponding frame.

To retrieve the stored features for training, a MongoDB API for Python called Pymongo is used. This acts as a server connecting the back-end database with the front-end webpage. Data retrieval can be roughly divided into four steps, namely establishing the connection, executing the query, iterating through the results and fetch each JSON object.

6. Gaussian mixture model

Once features are extracted from training data and stored into the database, we need to feed them into a machine learning algorithm to identify the patterns and make further predictions using the trained model. After comprehensive testing and evaluation, Gaussian mixture model was identified as having the best performance. Gaussian mixture model is a probabilistic model which assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters [9]. The model is commonly used as a parametric model of the probability distribution of continuous measurements or features in a biometric system [9], which suits our needs.

The model we use is implemented using the scikit-learn, which is an open source machine learning library for Python. It provides a collection of Python packages containing simple and efficient tools for data mining and data analysis. The packages are all open source and commercially usable, which allows for more scalability and customization. In the implementation of scikit-learn, the parameters of the Gaussian Mixture Model are estimated using the iterative Expectation-Maximization (EM) algorithm.

A Gaussian mixture model is a weighted sum of M component Gaussian densities, as shown in the formula below:

$$\mathrm{p}(\mathrm{x}|\lambda) = \sum_{i=1}^{M} w_i g(x|\mu_i, \Sigma_i)$$

where x is a D-dimensional data vector, $\lambda = \{w_i, \mu_i, \Sigma_i\}$, i= 1,. . . ,M, is a collection of all the parameters in Gaussian mixture models, $g(x|\mu_i, \Sigma_i)$, i= 1,. . . ,M, are the Gaussian distribution functions for different components, w_i, i= 1,. . . ,M, are the mixture weights, which must satisfy the constraint that $\sum_{i=1}^{M} w_i = 1$.

Each Gaussian density function has the form as below, where μ_i denotes a D-dimensional mean vector and Σ_i denotes a D×D covariance matrix.

$$\mathrm{g}(\mathrm{x}|\mu_i, \Sigma_i) = \frac{1}{(2\pi)^{D/2}|\Sigma_i|^{1/2}} \exp\{-\frac{1}{2}(x-\mu_i)'\Sigma_i^{-1}(x-\mu_i)\}$$

The whole Gaussian mixture model is defined by the parameters , $\lambda = \{w_i, \mu_i, \Sigma_i\}$, i= 1,. . . ,M, which is estimated using the Expectation-Maximization (EM) algorithm in the implementation of scikit-learn. This method is by far the most popular and well-established. Given the training data, the goal of EM estimation is to adjust the parameters so that the

likelihood of the GMM is maximized, which is in fact to find λ that can maximize the following probability:

$$p(X|\lambda) = \prod_{t=1}^{T} p(x_t|\lambda)$$

where $X = \{x_1, \ldots, x_T\}$, is a sequence of training vectors with length T. This is not an easy task as there is no closed-form solution, however, by introducing latent variables, the problem can be solved iteratively. To start with, random values for w_i, μ_i, Σ_i are chosen, then the membership of each training point is updated using the parameter values in the current iteration, i.e. assign training instance x_i to component k of our mixture models. Once we have reassigned the membership for all the training data, the parameters could be updated using the new membership, which again becomes an initial model for our next step. In each iteration, the model's likelihood value is guaranteed to increase monotonically and this process is performed until some convergence threshold is achieved.

Due to its capability of approximating a large collection of sample distributions, GMM is widely used in biometric systems, and to recognize animal species from acoustic data is one of its many applications. Each component in GMM is a spate Gaussian distribution function, with its own mean and covariance matrix. Together with additional weight parameters, GMM is good at forming smooth approximations for arbitrarily shaped densities. Take the histogram in Figure 6 for example, which plots the distribution of a single cepstral value from a 25 second utterance by a male speaker. The plot below the histogram is the GMM approximation with only one component, which is literally a Gaussian distribution. The third plot at the bottom show the approximation of GMM using 10 components, which is obviously more similar to the original histogram.

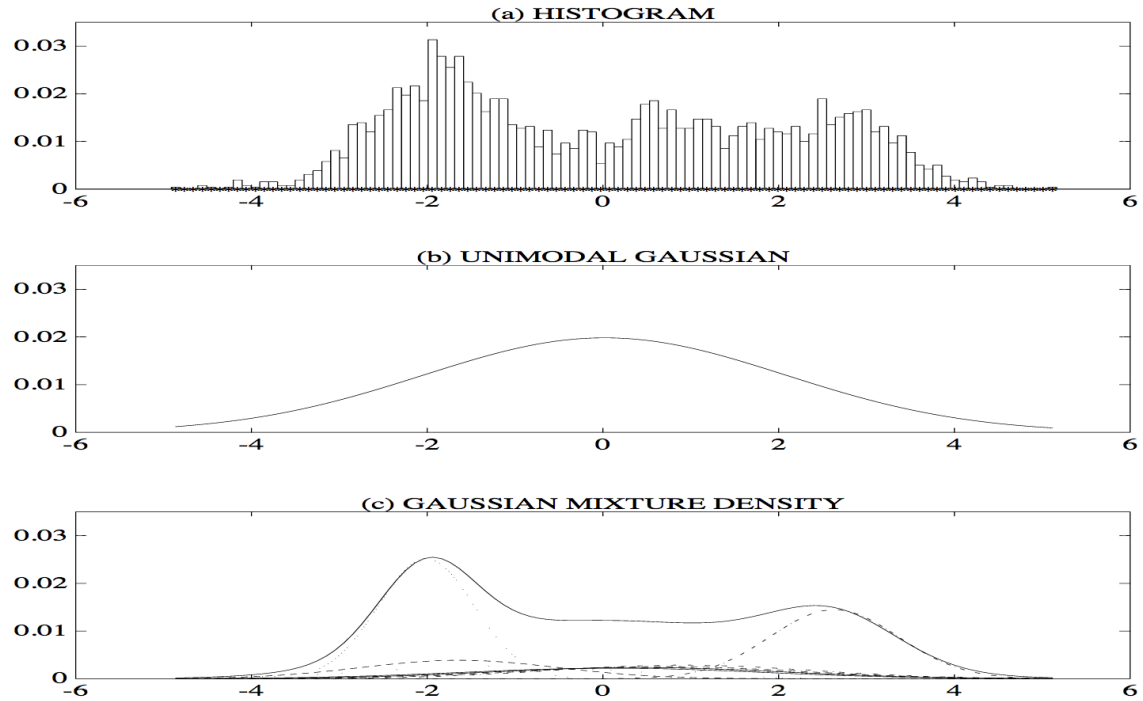

Figure 6. Modelling arbitrary density using Gaussian mixture model (b: 1 component, c: 7 components)

6.1 Gaussian mixture model usage details

In the previous part, the underlying structure and theory foundation of Gaussian mixture model was briefly introduced. Now we explain more about how GMM is applied in our project. In Figure 6, the feature data only has one dimension, which is represented by the x axis in the histogram, while the y axis represents the frequency. However, in our bird recognition project, the training features have 13 dimensions, and the MFCCs of all the audio files belonging to the same species are concatenated into a single matrix, with 13 columns in each row. A separate Gaussian mixture model is trained for each species to approximate the acoustic feature distribution, which means that if we are to recognize 6 species of birds, 6 Gaussian mixture models are trained. In order to better model the overall feature density, the number of components of GMM is set to 10 to keep adequate details and at the same time avoid overfitting.

Now that we have a Gaussian mixture model for each species, we are ready to proceed to prediction. The file to be predicted will be fed into each model, and the per-sample likelihood is calculated for each model. This is in the form of an array with each element corresponding to the likelihood of the short frame in the sound signal. Next the array of likelihoods will be summed up to represent the overall likelihood of the bird song file under a specific model, and bird species corresponding to the model with the highest likelihood will be the final prediction.

6.2 Prediction confidence

Once the classification model is trained, we are ready to predict the species of birds for users. However, for most machine learning models, the prediction comes with likelihood, that is, how confident the model is in the prediction. Because during prediction, we are actually trying to find the GMM (a weighted sum of Gaussian distribution components) which fits the audio file best, rather than finding a single Gaussian distribution component, the built-in likelihood of GMM in scikit-learn turns out to be not very useful [10].

In this project, we established a method to approximately model the confidence of prediction. In the prediction process, the sample scores are summed up for each model, and the model with the highest score sum is chosen. Assuming there are several models trained, and ranked according to the sums of their sample scores. If the gap between the highest-score model and second highest model is large, then it is likely we have made a correct prediction. However, if the gap is very small, it is possible that some small errors would totally change the final prediction, such as noise in the background of recordings, the difference in parameters of the model, or slight differences in the extraction process of features. This means that the scores of the top-ranked model and second-ranked model are most related to the confidence of our prediction, therefore the following formula is used to represent the confidence:

$$\text{confidence} = \text{second highest score} / (\text{second highest score} + \text{highest score})$$

As the sample scores are actually calculated on a logarithmic scale, the scores are all negative, so the above formula would always be larger than 0.5. The larger the ratio, the more confident we are in our prediction. Later from the prediction accuracy results, we will see that this confidence model matches the overall prediction accuracy well.

Another noteworthy function of our system is that when the model is very confident about a prediction, it will assume that the prediction is correct and add the new feature data into the database. This method is inspired by semi-supervised machine learning and allows the size of training data to grow over time. Currently the threshold is set as 0.7 manually, however this can be improved in the future by learning from data.

7. Prediction result analysis

The performance of our software varies depending on the number of species being predicted. Due to the limited amount of training data (about 200 recordings for 9 bird species), we used 7-fold cross validation instead of 10-fold to make sure there is adequate data for testing in each fold.

The prediction accuracy is a metric representing the ratio of correct predictions to the total number of predictions, it can be calculated from the following formula [11]:

$$\text{accuracy} = \frac{TN + TP}{TN + TP + FN + FP}$$

where TN, TP, FN, FP are the number of true negative cases, the number of true positive cases, the number of false negative cases, and the number of false positive cases respectively.

In addition to accuracy, the model is also evaluated using precision, recall and f1-score. Precision is the fraction of retrieved instances that are relevant, recall is the fraction of relevant instances that are retrieved, while f1-score is a combination of the two. They are calculated using the following formulas [12]:

$$\text{precision} = \frac{TP}{TP + FP}$$

$$\text{recall} = \frac{TP}{TP + FN}$$

$$\text{F1} - \text{score} = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

The 7-fold cross validation results for 5 bird species is shown in Figure 7. As we can see, the performance of our program varies with respect to the species of birds. The MFCC feature and GMM model seem to be most accurate for Southern Boobook, with f1-score reaching 0.80, much higher than the average f1-score 0.58. However, although the number of training files for Australian Golden Whistler is the largest among all the bird species, its performance remains poorest, with f1-score at only 0.44. This may be due to the fact that the feature extraction algorithm better differentiates Southern Boobook from other species. The general perception people have for machine learning applications is that the more training data there is, the higher the prediction accuracy will be. This may be true for the same bird species, if we could get more training data for Australian Golden Whistler, its performance is likely to improve. However, the feature extraction algorithm is also of great importance to the overall performance.

	precision	recall	f1-score	support
Australian Golden Whistler	0.53	0.38	0.44	48
Brush Cuckoo	0.47	0.76	0.58	21
Grey Shrikethrush	0.52	0.52	0.52	29
Southern Boobook	0.71	0.92	0.80	13
Spangled Drongo	0.70	0.66	0.68	50
avg/total	0.59	0.58	0.58	161

Figure 7. Predictions of 5 bird species with accuracy 0.584

The results of predictions for 6 bird species and 9 bird species are shown in Figure 8 and Figure 9 respectively. It is evident that the prediction accuracy decreases as the number of bird species grows. When there are 9 bird species, the accuracy declined to only 0.516, and even lower accuracy for predictions of more than 9 bird species can be expected. The reason for this is straightforward, as the number of species goes up, it becomes much more difficult to differentiate between various species, and the same rule applies to human as well.

Another interesting phenomenon worth noticing is that the relative performance of predictions for each bird species remain stable. For example, in Figure 8, the accuracy of the Australian Golden Whistler is much worse compared to the Southern Boobook, and this pattern continues in Figure 8 and Figure 9, where the F1-score of the Australian Golden Whistler remains about half of the Southern Boobook. This further proves our previous assumption that the feature extraction algorithm differentiates some bird species better than others, and this difference remains the same even when the amount of training data changes. To show this pattern better, a bar chart is drawn in Figure 10, where we can see that the relative performance of different birds remains similar, or in other words, the bird we are good at predicting using data of 5 bird species, continues to give better results using data of 6 bird species or 9 bird species.

	precision	recall	f1-score	support
Australian Golden Whistler	0.53	0.38	0.44	48
Brush Cuckoo	0.47	0.76	0.58	21
Grey Shrikethrush	0.52	0.52	0.52	29
Southern Boobook	0.71	0.92	0.80	13
Spangled Drongo	0.70	0.66	0.68	50
avg/total	0.59	0.58	0.58	161

Figure 8. Predictions of 6 bird species with accuracy 0.577

	precision	recall	f1-score	support
Australian Golden Whistler	0.47	0.30	0.36	47
Australian Magpie	0.64	0.50	0.56	28
Brush Cuckoo	0.62	0.68	0.65	31
Eastern Whipbird	0.26	0.29	0.27	21
Grey Fantail	0.35	0.54	0.42	13
Grey Shrikethrush	0.45	0.47	0.46	32
Pied Currawong	0.45	0.50	0.47	18
Southern Boobook	0.59	0.93	0.72	14
Spangled Drongo	0.65	0.65	0.65	48
avg/total	0.52	0.52	0.51	252

Figure 9. Predictions of 9 bird species with accuracy 0.516

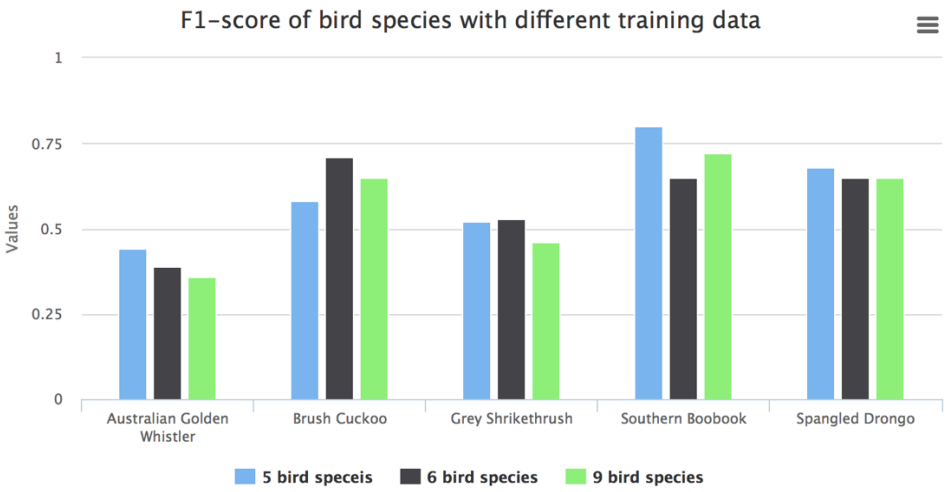

Figure 10. Relative performance remains similar in different training data

8. Speed up

When there are only small modifications to the database, for example the database is not updated or only several files' features are added, there is no need to train a new GMM every time users predict species with audio files. That is to say, the model can be trained once, stored on the server for several days, and applied to all the new audio files during this time period. This method greatly saves the waiting time of users as well as the computing resource on the server. For frequent user accesses, the model can be updated everyday or every week, while if there are not many users, the model can be updated every month.

This function is implemented with the "pickle" package in Python [13], as its name suggests, it saves Python object onto the disk for future use.

9. Front end

All the functions are finally shown as a web application. Flask, a lightweight web framework, is used as the connection between Python and the web front end. On the webpage, users can choose to update the prediction model to the newest version, or if they do not want to wait for the training time, they can choose to only upload the audio file and the prediction result will be shown. In addition, since it is useful to capture location information during the recording, we can include it in the name of the recording, like

name_-37.5_144.8_.wav

and the location can be visualized [14] on the web page as shown in Figure 11.

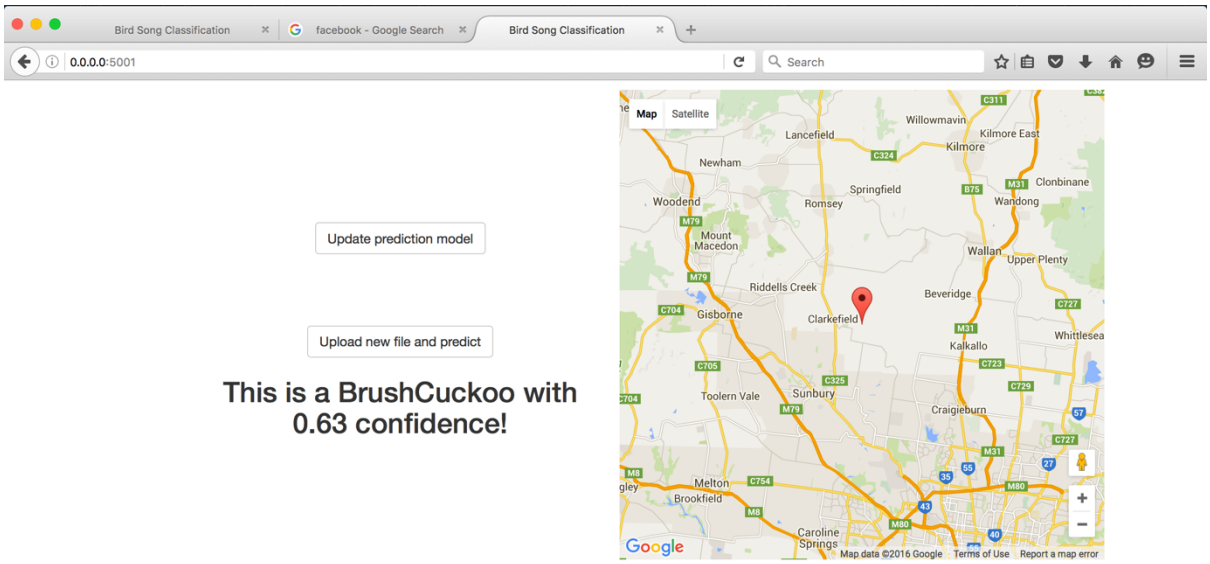

Figure 11. Webpage after prediction.

10. Conclusion

In this project, the focus was to predict species of birds from audio data. To do this, I used MFCC for feature extraction and GMM for machine learning. The prediction accuracy varies depending on the specific species being predicted. As the amount of training data was very limited, we only investigated the prediction for 9 bird species. If we could have access to more audio data in the future, it would be probable to not only improve prediction accuracy for each existing bird species in the system, but also increase the number of species that are predicted. The user interface could also be in the form of a mobile app, which allows location information to be automatically saved during recording. The future work may be to include location information in the prediction model, and to improve audio quality by signal processing.

Reference:

[1] Miguel A. Acevedo, Carlos J. Corrada-Bravo, Héctor Corrada-Bravo, Luis J. Villanueva-Rivera, T. Mitchell Aide, Automated classification of bird and amphibian calls using machine learning: A comparison of methods, Ecological Informatics, 20 Jun 2009.

[2] Anuradha Abewardana, Upul Sonnadara, Classification of Birds using FFT and Artificial Neural Networks, Proceedings of the Technical Sessions, 28 (2012) 100-105.

[3] Steven B. Davis, Paul Mermelstein, Comparison of Parametric representations for monosyllabic word recognition in continuously spoken sentences, IEEE Transactions On Acoustics, Speech, And Signal Processing, 4, Aug. 1980

[4] Mel Frequency Cepstral Coefficient (MFCC) tutorial, http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/

[5] Róisín Loughran, Jacqueline Walker, Michael O'Neill, Marion O'Farrell, The Use of Mel-frequency Cepstral Coefficients in Musical Instrument Identification, International Computer Music Association, Aug. 2008.

[6] Tahira Mahboob, Memoona Khanum, Malik Sikandar Hayat Khiyal, Ruqia Bibi, Speaker Identification Using GMM with MFCC, IJCSI International Journal of Computer Science Issues, Mar. 2015

[7] Jonathon Jongsma, Xeno-canto API 2.0, http://www.xeno-canto.org/article/153

[8] MongoDB, https://en.wikipedia.org/wiki/MongoDB

[9] Douglas Reynolds, Gaussian Mixture Models, http://www.ee.iisc.ernet.in/new/people/faculty/prasantg/downloads/GMM_Tutorial_Reynolds.pdf

[10] sklearn.mixture.GMM, http://scikit-learn.org/stable/modules/generated/sklearn.mixture.GMM.html

[11] https://en.wikipedia.org/wiki/Accuracy_paradox

[12] https://en.wikipedia.org/wiki/Precision_and_recall

[13] pickle — Python object serialization, https://docs.python.org/2/library/pickle.html

[14] Google Maps API, https://www.google.com.au/work/mapsearth/