University of Sheffield
Department of Computer Science

Masters Project Dissertation

Bird Song Recognition using GMMs and HMMs

Author:
Douwe Gelling

Supervisor:
Dr. Phil Green

This report is submitted in partial fulfilment of the requirement for the degree of MSc in Computer Science with Speech and Language Processing

September 1, 2010

All sentences or passages quoted in this dissertation from other people's work have been specifically acknowledged by clear cross-referencing to author, work and page(s). Any illustrations which are not the work of the author of this dissertation have been used with the explicit permission of the originator WHERE POSSIBLE and are specifically acknowledged. I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this dissertation and the degree examination as a whole.

Name: Douwe Gelling

Signature:

Date: September 1, 2010

Abstract

The importance of using temporal information in recognising bird vocalizations is investigated. Part of the methods used by the SongScope software are replicated and a set of experiments is carried out, where syllables and songs are recognised using GMMs. The results indicate that there might be a small benefit to using more states in a HMM than using more mixtures, however this difference is very small and might be due to variation. Furthermore, other techniques appear to have much greater influence, such as using multiple models for the syllables of single species and recognising whole songs instead of only syllables. These increases in accuracy form a clear trend, but variation was so high that the possibility remains it is due to chance variation.

Contents

List of Figures

List of Tables

Abbreviations

Abbreviations	- Long form
DTW	- Dynamic Time Warping
GMM	- Gaussian Mixture Model
HMM	- Hidden Markov Model
SVM	- Support Vector Machine
MFCC	- Mel-Frequency Cepstral Coefficient
E-M	- Expectation-Maximization
FFT	- Fast Fourier Transform
DCT	- Direct Cosine Transform

1 Introduction

In ornithology the use of audio recordings is wide-spread by both professionals and amateurs. Not only are they used as reference for manual recognition of bird sounds heard in the wild, but inspection of spectrograms by experts is a very popular technique, dating back to the 1950's. There are many applications for bird vocalization analysis. For example, ornithologists may be interested in finding out whether a particular, perhaps rare, species has appeared in a given region. In such cases, recorders might be left in several places in the region for a period of time, to be analysed later to see whether the vocalisations of that species appear. Another might be to find out how large the population of a particular species is, or even to find out how many different bird species occur in a given region.

However, such techniques are labour-intensive, and generally require experts, who can identify the sounds of different species on a spectrogram. The cost of training these experts can be high, and the humans have varying ability in detecting certain birds. One expert might be better at recognising certain birds than others. Also, birds may be less inclined to sing if a human is present. This is of course only a problem if a human is actually present on site to do the census, and is not listening to a recording. [8]

The cost of using experts is especially problematic, as there generally is a large amount of lengthy recordings to be inspected. This can make the use of recordings even more expensive and time-consuming than actually sending experts into the field, as the recorders have to be placed and picked up, and after that listened to. [8] This task could benefit greatly from using automated machine-learning techniques and in recent years research into doing this has increased. Usually, techniques from Automatic Speech Recognition or Automatic Speaker Recognition are appropriated to do this.

However, as of now, no common framework has been proposed to do this research. What's more, research into the issue of recognising bird song has not gained much traction in the scientific community. What research there has been has generally used testing databases of the researcher's own devising, making comparisons between different studies that have been done in this field hard.

Also, these studies have not always focused on the task of recognising bird song itself, but sometimes more on showing that a particular machine learning technique can be used. The studies that have been done have also generally taken techniques that are often used for human speech recognition and tried to show that these can be used for recognising bird vocalisations.

Most of these studies have researched methods that don't take the temporal aspect into account, but modelled the average characteristics of individual syllables. In those studies where temporal information is used, for example in studies using Hidden Markov Models (HMMs), often the result would be that the methods that do not take temporal information into account would perform better than those that do. This is curious, because although the ordering of syllables on the song level and even the syllables that are used might be dependent on the bird itself, it would seem that the temporal structure of individual syllables does have impact. For example, it is easy to imagine two syllables that have the same spread in

frequency, but where one goes from low to high and the other from high to low.

Furthermore, one of the few software packages available today that claims to be able to do bird song recognition (SongScope), does so by using HMMs, and explicitly modelling temporal structure. Most of the techniques used in the literature also try to model different bird songs or individual syllables in a statistical manner. Contrary to this, there is some evidence in the literature to suggest that birds remember specific syllables from other birds, and recognise these individual syllables. It might be that this is also how they recognise the songs of birds they know [7]. Regardless of this, in the existing literature, it seems that statistical methods do work better than template-based methods such as Dynamic Time Warping (DTW).

Another method that is employed by the SongScope software, is to model different syllable groups from a single species, by clustering the training data beforehand. This approach has also been used in scientific literature, but mostly through the use of codebooks, where unlike with temporal structure, results did improve. However, these grouping methods were mostly used in combination with models that did not incorporate temporal information.

The current research discussed here will make use of statistical methods, and try to find out if temporal information in syllables of birdsong do in fact increase recognition accuracy of bird vocalisations. Also, a clustering method will be used, to see what the gain is when combined with models using temporal information. The models that will be used in this research are GMMs as well as HMMs.

In the next sections, first a brief overview of the properties of bird song is given, then the existing literature is reviewed, as are the methods used by SongScope. After that the current research is outlined, the used methods will be detailed, and the results of experiments on songs from 5 different species will be given. Lastly, the results will be evaluated and conclusions will be drawn.

2 Bird vocalisations

Bird vocalisations are usually divided into two categories: calls and songs. Generally, calls are used for some form of communication and are usually short, such as alerting other birds of a predator, whereas songs are mostly used to attract mates and can be quite long. Bird songs have a clear structure, and can be subdivided into phrases, syllables and elements. Often, syllables will consist of a single element, but sometimes they are more complex. Phrases are usually made up of repetitions of a syllable, and the song itself is a series of phrases.[5] An example of the subdivision of birdsong into smaller parts can be seen in Figure 2.1, where the song of the common chaffinch is displayed.

This can be seen as analogous to human speech, where the basic building blocks are phones, instead of elements. In human speech, syllables are groups of phones, usually consisting of a vowel, which is surrounded by 0 or more consonants. Words are composed of syllables, and generally carry meaning on their own. For birds, we could go on to treat syllables, phrases and songs as we would do with human syllables, words and sentences. That is, a syllable for bird vocalisations is analogous to a syllable in human speech, a phrase in bird vocalisations is analogous to words

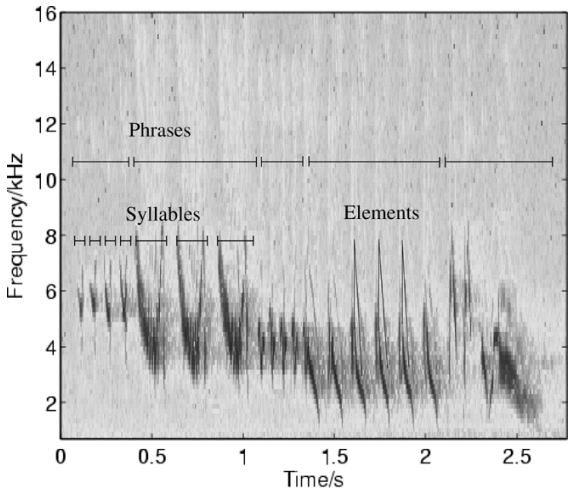


Figure 2.1: Subdivision of birdsong of the common chaffinch into phrases, syllables and elements. Figure taken from [11]

in human speech and songs are analogous to sentences.

The subdivision of bird vocalisations is arbitrary though, since many bird vocalisations do not carry meaning, and certainly not in the way human speech does. The subdivision is made entirely by humans, as a matter of convenience. Also, in contrast to human speech, there are often physical pauses between syllables, whereas in human speech, normally there aren't any pauses between words, or even sentences. This is quite visible in Figure 2.1. This is not always the case, however, as there are some species that can generate continuous song via a complicated breathing pattern [11].

The analogy with speech and language recognition breaks down further, with the acquisition of the training data. In human speech, we know what the words and phones are, and what's more, we can ask different speakers to produce specific words. We can also ask human speakers to talk into a microphone, to obtain high-quality data. For birds, we are much more dependent on chance, having to wait for the bird to make a sound. Also, we have to guess at what the syllables and phrases of the vocalisation are. Because a lot of recordings have to be made in the wild, they are generally of low quality, with a low Signal-to-Noise Ratio, due to background noise made by other animals, the wind and human sources. Also, the signal usually has to travel quite far and reverberates off the trees. In Automatic Speech Recognition, it is very hard to do recognition on data captured with far-field microphones than on data captured with close-talking microphones. The same holds for recordings which contain reverberated speech.

Lastly, the spectral properties of bird vocalisations are quite different from that of human speech. Bird vocalisations are generally very tonal, consisting of either a single frequency at a given point in time, or of a frequency and a few harmonics. This means most of the energy is located in a narrow region of frequency, whereas in human speech, the high energy regions are often distributed over a larger spectral range, due to fricatives and the formants in vowels. Also, birds can produce sounds with a structure in the lungs, called a lyrinx, which allows them to produce two

different tones at the same time [5]. These difficulties are not insurmountable, however, and given the considerable amount of money that can be saved by doing bird song recognition automatically, it is clearly of importance to tackle these problems.

3 Related Work

The related work in the scientific literature will be reviewed in this section, as well as the commercial implementation from SongScope. Because many articles in the scientific literature employ some well-known techniques, these techniques will briefly be discussed first, after which the literature is reviewed.

3.1 Commonly used techniques

3.1.1 Dynamic Time Warping

Dynamic Time Warping (DTW) is a technique for recognising time-variant data. The general idea is to build a database of labeled reference samples, and when recognising a test sample, try to find the reference sample that is closest to the test sample. This can be visualised by constructing a trellis, with the time framse of one sample on the x-axis and of the other on the y-axis. The goal is to find the least costly path through the trellis, starting at the bottom left and ending at the top right. Each possible move (go upwards, to the right or diagonally: to the right and upwards) has a cost associated with it, and moving diagonally is usually the least costly. For every point in the trellis, the distance between the two feature vectors associated with the corresponding time frames in the input samples is calculated and added to the cost. This way, a compromise is found between a simple path (as many diagonals as possible) and matching the data. For finding the best path, the Viterbi algorithm can be used.

3.1.2 Gaussian Mixture Models

A Gaussian Mixture Model (GMM) tries to model a class by using n-dimensional Gaussian distributions, where n is the dimensionality of the input data, generally with a diagonal covariance matrix. It often uses more than one gaussian to represent this class, because the distribution of points might be of a shape that can't be modeled by a single gaussian. For example, in the 2-dimensional space, the data of a class might have a distribution in the form of an L. As a 2-dimensional gaussian can only model circles and ellipses, the only way to model this with one gaussian is to make it a large ellipse. By using multiple gaussians to model the L, one could represent the data as two thin ellipses, which would give a more accurate representation. This is illustrated in Figure 3.1, where on the left, only one gaussian is used, and the gaussian describing the blue class, has to be so large as to cover the datapoints of the green class as well.

When training a GMM with N mixtures, the problem is that it is unknown at first which data points belong to which mixture. To solve this, k-means clustering

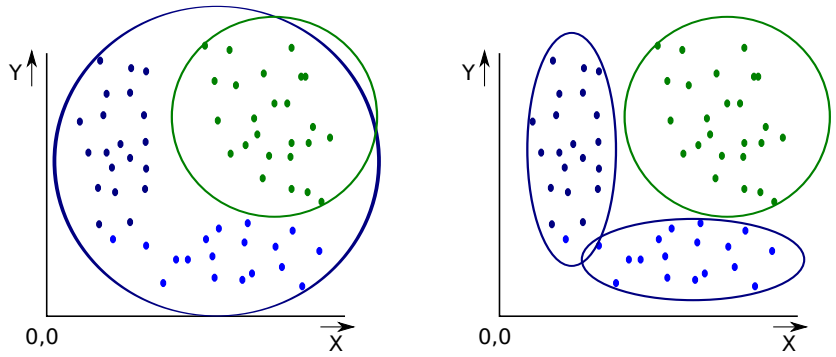


Figure 3.1: Left: modelling 2D data when only one Gaussian is used. Right: modelling 2D data when up to 2 Gaussians are used.

can be used to initialize the GMMs, after which Expectation-Maximization (EM) training can begin. Recognition on GMMs is done, by training one GMM for each class, then computing the (average) probability of all frames given a class, and the class that gives the highest probability is the class that the test sample is assigned to.

3.1.3 Hidden Markov Models

A Hidden Markov Model (HMM) is a directed graph, each node of which represents a state. In speech recognition, these graphs are usually left-to-right, and each state has an arc to itself. These arcs have an associated probability. HMMs are used to model the temporal properties of a signal. Each state in an HMM has an associated probability function, modeling the probability of observing a particular feature vector in that state. Usually, these probability functions are represented by GMMs.

HMMs compute the probability test samples by starting in one state, computing the probability of observing the first feature vector, then going to the next state (which may be the same state), computing the observation probability for that state, and continuing until all feature vectors of a sample are observed. In Figure 3.2, the general layout of HMMs is illustrated. The HMM has 3 states, a begin-point and end-point. All states have a probability of going to the next state, $Am-n$. In left-to-right models, there is only one possible begin-state, so the beginpoint has only one arc and $A0-1$ is 1, but this is not the case in all HMMs. The probabilities of arcs going from a state have to sum to 1. Furthermore, in simple models, the probability of going to the end state is 0, but if an HMM is comprised of several smaller HMMs, this is not always the case.

The problem now becomes to assign the right state to every feature vector.This can be solved by constructing a trellis, with the observation feature vectors on the x-axis and the possible states on the other. When using a left-to-right model, the possible moves are to go right or to go diagonally (right and upwards). Knowing this, the optimal path has to be found, starting in the bottom-left and ending in the top-right. For this the Viterbi algorithm can be used as well.

Training is a bit trickier, as it is not known which feature vectors belong to

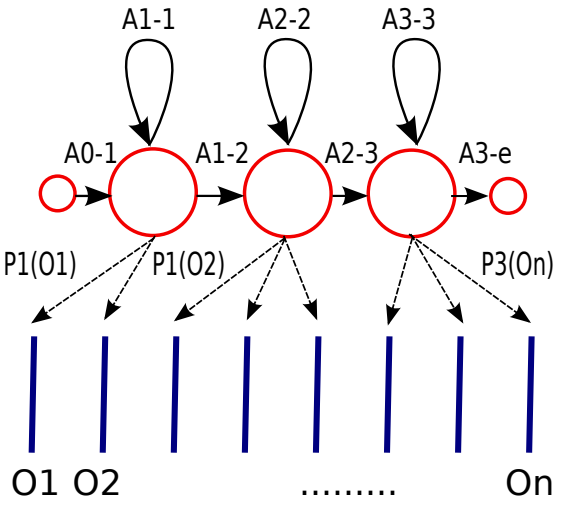


Figure 3.2: Diagram of an HMM, with transition probabilities $Am - n$ and observation probability function $P(O)$.

which state. To solve this, the training samples can be evenly divided among the N states, the first $\frac{1}{N}$ feature vectors being assigned to the first state, the second $\frac{1}{N}$ feature vectors assigned to the second state and so on for every training sample. After this, the GMM of a state can be trained as normal on the data assigned to that state. Following this, EM training can be done, by iteratively using the viterbi algorithm to assign feature vectors to a state and then training the GMMs and recalculating the transition probabilities on that data. Recognition is done by training an HMM for every class, computing the probability of a test sample for the HMMs of every class and assigning the class of the HMM with the highest probability to the test sample.

3.1.4 Support Vector Machines

A Support Vector Machines (SVM) is a classifier, which is used to classify data into one of 2 classes. The input data are vectors of N dimensions, and the training data can be mapped into N-dimensional space. For SVMs, the idea is to linearly separate the data of 2 classes using a hyperplane, which is constructed from representative data points of both classes, that are close together. Such a hyperplane cannot always be constructed, and in those cases a kernel function is used to map the data from N-dimensional space into $N + 1$-dimensional space, and an attempt is made again. To prevent this from becoming too computationally expensive, often a certain proportion of the data, that is inconsistent with the hyperplane, is allowed to be ignored.

An example is given in Figure 3.3, where in lower dimensions, data is distributed too complexly, so a linear separation can't be made, but im higher dimensions, this is possible. The hyperplane is constructed as the plane halfway between representative samples. These samples are the samples closest to the hyperplane from either class, and the samples are chosen such that the distance of the samples to the hyperplane is maximized.

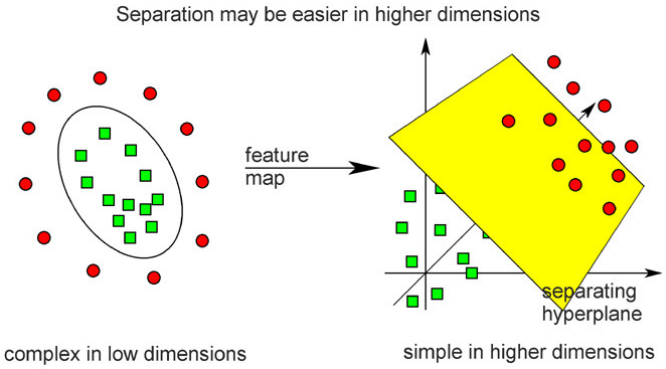


Figure 3.3: Principle of using SVMs, in low dimensions (left), data can't be linearly separated, but in higher dimensions (right) they can. Figure taken from `http://ccforum.com/content/11/4/R83/figure/F1`

Since this technique can only differentiate between two classes, multiple SVMs have to be constructed to do recognition on more classes. This can be done by constructing an SVM for every pair of classes that exist, and then using a decision hierarchy to decide which class is the right class, or by constructing an SVM for every class, where the other class data is the combined data of all other classes.

3.2 In the literature

Firstly articles that discuss among others, techniques that make use of temporal information will be discussed. After that the remaining articles will be discussed.

3.2.1 Kogan & Margoliash

One of the first studies into bird song recognition was done by Kogan and Margoliash [9]. They compared a DTW method with a HMM method for recognising the songs of two species, zebra finches and indigo buntings. These two species differed significantly in the size of the spectral band that their vocalisations have. Hidden Markov Model Toolkit (HTK) was used for making the HMMs.

In the DTW approach, the system labels and segmetns the input data by comparing the input with a set of prespecified template patterns [9]. In this technique, the temporal frames of the input and those of the templates are organised in a 3-dimensional lattics (i, j, k). Here, i indicates the index of the frame of the input, j indicates the index of the template frame and k are the numbers of the individual templates. It makes use of a distance metric, that calculates the distance between the two multidimensional vectors of the input signal at time fram i, and that of template k at time-frame j [9] This distance is the dynamic time-warping distance between the two multidimensional vectors.

It therefore tries to find the optimal sequence of template patterns, given the input. The DTW implementation that was used in this study was constrained not do time-warping at more than a factor 2. With this method, only variability in the temporal comain is accounted for, however, and not variability in the spectral domain.

The HMM method was developed using HTK. This toolkit is mostly dsigned to do human speech recognition, but it was adapted to do the birdsong recognition task [9]. Three different categories were modelled by an HMM, calls, syllables and cage noises. They mostly used left-to-right models with five states. Only the middle three states were used to emit signals, the outer two were used to facilitate transitions between different models. The emition probabilities were modelled using GMMs.

For recognition, the Baum-Welch algorithm was used to find the optimal state-sequence. Training samples were selected at random, but such that all vocalisation classes were represented in the training set. Training files with uncommon elements were specifically included for training. During recognition, the token passing implementation of the Viterbi algorithm was used.

All of the training and testing data was either fully or partially labeled by hand. The labels were cross verified and thus deemed to be highly reliable. The templates for the DTW approach were manually selected, from the beginning, middle and end of recordings. Features for the DTW approach consisted of the frequency bins of fourier transforms of the data, where the bins below 500 Hz were left out, to remove low-frequency noise. Thus, the frequency bins reached from 0.5 to 10 kHz. The FFT size, frame rate and minimum duration for elemanets to be recognized were detemined experimentally, for every bird separately [9].

For the HMM tests, six different parametrization types common in speech recognition were analyzed using HTK. These included MFCCs and LPCCs. It turned out that MFCCs worked best for the bird recognition task, so it was used along with the first and second derivatives. Window size and window overlap were different for either species and chosen experimentally. Both techniques were tested against continuous recordings of all animals in the database, including the training sets.

Both techniques turned out to perform nearly equally well, but they were sensitive to noise. The DTW-based approach in particular needed many more templates to be able to deal with noisy environments. By adding extra templates for cage noises, the HMM-based model could be outperformed by DTW, but this did necessitate a lot of extra work. In general, for both bird species, both models were relatively good at recognising different syllables, although the HMM models were a little more robust. Also, recognition could be improved for short syllables, by setting the amount of states to be 4 instead of 5.

3.2.2 Somervuo et al.

Somervuo et al. expanded on the previous work, by comparing 3 different feature representations fr use in DTW, HMMs and GMMs in the task of bird species recognition. They used MFCCs, sinusoidal models, which had earlier generated good results for tonal bird sounds and vectors of descriptive features [11].

The bird vocalisations were segmented into syllables by computing the smooth energy envelope of the signal, and selecting the regions that are above a syllable threshold. The threshold is computed by first setting it as half the estimate of the background noise level, which is initialized as the global minimum. It is then

iteratively updated by finding gaps between syllables, setting the background noise level according to the energy in those gaps, and setting the syllable threshold to half that. This is repeated until fairly stabilized.

Once syllables are found, they're grouped together if they are very close to each other (less than 15 ms apart) [11]. The automatic segmentation was compared to a manual segmentation of 50 songs, and it was found to detect about 93% of all syllables. However, there is high variablility between species.

The sinusoidal representation was made by first taking the FFT of the signal with a hanning window of 256 samples and 50% overlap between frames. An FFT of size 1024, with zero-padding was then computed. The audio was sampled at 44.1 kHz. After this, the frequency component with the maximum energy for that frame is chosen. From that frequency component, the phase and magnitude of the optimal sinusuidal pulse is found. Because some types of songs also have harmonics, and the component with the highest energy is not always the base frequency, but sometimes one of the first 3 harmonics, three different classes were added, each having the component with the highest frequency as a different harmonic. The vocalisations were automatically put into one of the four classes, with relatively high accuracy [11].

The MFCC vectors were computed on 20-ms windows, with 10-ms intevals, giving 8-dimensional MFCCs, and versions were used with the first, as well as both the first and second derivatives. The derivatives were computed over a time-span of 50 ms. The descriptive parameters consisted of several features, which were computed over the entire syllable. Some features were computed over a set of consecutive frames of the syllable, with a frame size of 256 samples, and 50% overlap. The means and variances of those features were then used in the descriptive parameters. The descriptive parameters were only used in comparisons wit other fixed-length descriptions.

First, DTW-based tests on recognising single syllables of different species were carried out using the sinusoidal model using several distance metrics and one using MFCCs with and without derivatives, using euclidean distance. The MFCC-based representations did slightly better, but adding derivaties did not help. Interestingly, it was investigated whether feature representations of fixed dimensionality could be used instead of DTW, and these turned out to work roughly equally well on single syllables [11]. The feature representations of fixed dimensionality were obtained by projecting the sinusoidal trajectories into the fixed-dimensionality features, or by averaging MFCC values.

HMMs and GMMs were later used in a test where multiple consecutive syllables had to be recognised. There was one HMM or GMM to represent an entire species. Diagonal covariance matrices were used in both. HMMs and GMMs were both tested on recognising song fragments and later on entire songs. For both tests, they semed to perform equally well, but both did much better than when recognising single syllables and did better on recognising a whole song than on recognising a song fragment.

3.2.3 Lee et al.

In the study by Lee et al., HMM performance is compared to that of the average LPCCs and MFCCs of single syllables, on two databases of 420 and 561 bird songs. If a codebook was made with several representative models, this could significantly outperform the HMMs.[10]

The input songs were segmented, by first computing the short-time fourier transform (STFT), setting n at 0 and then finding the point (f_n, t_n) in the fourier transform with the greatest magnitude. The position of the n-th syllable is set to be t_n. The amplitude (in dB) of the n-th syllable at that point is computed, and compared with the amplitude of the first syllable. If the difference between these amplitudes is greater than a threshold β, the syllable extraction process is halted. The kind of window, window length and overlap that was used was not mentioned.

Otherwise, starting at t_n, the peak is traced backwards and forwards through the STFT, again stopping if the difference between the amplitude of the syllable at point t_n differs from that at point $t_{n-\tau}$ by more than the threshold β. This will find the start and end points of the syllable. The magnitude of all points between and including the start and end times is set to 0, and the process starts overnew, but n is set to be $n + 1$ [10].

First, four different feature representations are investigated. One is the average of the LPCCs over the all frames over all syllables, another is the average MFCCs over all frames of all syllables. This is done by first computing the average values for every syllable, then normalizing the average values for that syllable between 0 and 1, over the range of values the features take within a syllable. This is done to compensate for the different range of magnitude the different syllables have. After this, the features of all syllables of a species are averaged, to obtain the representative feature vector for that species. The LPCCs and MFCCs consisted of 15 coefficients.

Because birds generally have a set amount of syllables that they use, another option is investigated. This is to not take a single feature vector as representation of the entire species, but to take a set of representative feature vectors, each representing a different syllable, called a codebook. The different feature representations are obtained by clustering the feature vectors and averaging the values for every cluster as described above.

The clustering is done, by an iterative algorithm over the set of all feature vectors S for a species:

1. Set the current cluster to be represented by the first feature vector in S, remove this feature vector from the set S

2. Find the nearest feature vector in the set to that cluster and remove it from the set. If the distance between this feature vector and the cluster is larger than the threshold T_d, go to 4.

3. Add the current feature vector to the current cluster, recalculate the representative of the cluster to be the average of all features in the cluster, go to 5.

4. Add the current cluster to the list of clusters, create a new cluster with the current feature vector as its representative

5. If the set is empty, return the list of clusters. Otherwise go to 2.

Unfortunately, it is not mentioned which distance metric is used, but presumably this is the euclidean distance. Another technique that was investigated, was Linear Discriminant Analysis (LDA). This technique tries to minimize the within-class distance while maximizing the between-class distance, and thus improve recognition accuracy. The idea here is to reduce the features space from n dimensions to d dimensions, where $d \leq n$ [10].

During recognition, the test signals are segmented into syllables using the same technique as during training. The average LPCCs and MFCCs are then calculated for every test syllable, and if necessary, the same linear transformation is, applied to this feature vector as to the reference feature vectors. The test syllables are classified, by finding the reference feature vector with the smallest distance from the test feature vector, using the euclidean distance. The class of this reference feature vector is then taken as the class of the test vector.

The techniques were tested on two different databases, containing 420 and 561 bird songs respectively, each corresponding to a single species. The recordings were all segmented, and half the resulting syllables were used for training and the other half for testing. The MFCC and LPCC techniques were then compared to each other, and a reference HMM system. Unfortunately, the amount of states and layout of the HMM is not reported, neither is reported what was used as feature vectors for the HMMs, or if there was one HMM for every species or one for every syllable type (using the same clustering as for the codebook approaches, for example).

First, the HMM approach was compared to the average MFCC and average LPCC methods, both with and without LDA. The HMMs performed about equally well as the average LPCCs, and for both MFCCs as LPCCs, using LDA gave a significant improvement. The average MFCC with LDA approach was most successful here.

In the second experiment, the Codebook approaches were investigated, also with and without LDA. These turned out to work much better than the averaging approaches, reaching up to 87% accuracy for the MFCC + LDA configuration, whereas the corresponding approach using averages got only 68%. LDA in all cases reduced the feature space to either 11 or 12 features [10].

Unfortunately, due to the setup of the experiment, it could be that the HMM was simply set up in a bad way, or was using FFT values as feature vectors. Also, because the training and testing data was taken from the same recording, it might be that the models don't generalize well over different amounts of background noise or different birds from the same species.

3.2.4 Tyagi et al.

In their 2005 paper, Tyagi et al. compared using Speactral Ensemble Average Voice Prints (SEAV), GMMs and DTW for the task of identifying 15 different bird

species. The SEAV technique was a new technique that was introduced by Tyagi et al. [12]

Before computation of the SEAVs, the input is windowed with a Hanning window, with a width of 20 ms and an overlap of 10 ms. The size of the FFT was 1024 frames, the sample rate of the input varied between 8000 and 44100 Hz. The SEAVs of the input was computed by, for each bird call signal, taking the sum of the values in all frames for every frequency bin. This results in a vector of N/2 - 1 frames for an N-point FFT, for every bird call signal.

The recordings were segmented by using energy and zero crossing rate to separate the bird calls from regions of silence and background noise. After segmenting, multiple bird call templates were generated for every recording. During testing, these templates are then compared to the test signal, for which an SEAV is also computed. The species of the reference SEAV for which the euclidean distance to the test SEAV is the lowest, is the recognized class.

The technique is compared with a DTW technique and using GMMs. For the DTW technique, the spectrograms of bird calls were used as the reference templates, and the euclidean distance is used to determine similarity. In the GMM technique, several different methods to calculate feature vectors were investigated. This included MFCCs, Perceptual LPCCs and Relative-Spectral Perceptual LPCCs. The dimensions of these feature vectors varied from 6 to 13.

Tests using these techniques were done on recordings from 15 different species. The results of the first test that was done, is that SEAV did better than the DTW technique and the GMMs using LPCCs, but the GMM using MFCCs did best, scoring 100% accuracy. Unfortunately, it isn't mentioned how the test signals were obtained, nor where the recordings were made.

The last techniques that were tried, was to combine the SEAV and DTW techniques at rank and at measurement level. In the first option, both techniques would be used to recognise a test signal, and the top three ranking reference templates for either techniques are further considered. Every bird in these two sets of reference templates was assigned a score based on its rank. If a bird occurs twice (once in each set), its total score is equal to the sum of its two scores [12].

The technique that combined the systems at measurement level, did so by considering the first three euclidean distances output by either system. If a bird occurs in the output of both systems, the scores would be combined by taking the mean of the weighted sum of the normalized euclidean distances. The bird with minimum score would be the bird that's recognised in both techniques.

Finally, a method to combine the two techniques was investigated that combined them at rank level at first, as in the first method of combining, but if a tie occurred, the measurement level combination was done for only those birds. It is not mentioned how ties are broken in the original method.

By using the first method, an improvement of almost 6% could be gained, but the second method on its own led to a decrease equal to the increse of the first method. Lastly, the third method managed to increase performance by just over 11%.

In conclusion, these methods seem to work fairly well for bird calls, but unfortunately no recognition is done on bird song. Also, it is unfortunate that no

mention is made of what the test set consists of, as it may well be these techniques are sensitive to noise or changes in which individual bird is doing the singing.

3.2.5 Fagerlund

In his 2007 study, Fagerlund compares two different representations, MFCCs and descriptive parameters in an experiment on recognizing syllables of vocalisations of birds. Two different datasets were used, with 6 and 8 different species respectively. The used algorithm was an SVM, and it was compared to a nearest neighbour classifier [6].

Bird songs were segmented by computing a smooth energy envelope of the signal, then estimating the noise level as the minimum energy in the signal. The initial threshold is set to half that, after which an iterative algorithm is run, defining anything above the threshold as syllables, updating the noise estimate from the average energy level in the gaps and setting the threshold to half that level. The algorithm is run until it converges on a solution.

The used features are MFCCs, MFCCs with first derivative and MFCCs with first and second derivatives as well as descriptive parameters, similar to those used in [11]. The support vector machines are trained on pairs of birds and run against each other in a pyramid setup. The results of recognition on the two datasets of the SVMs using the four different representations, and a mixture using all MFCC features and the descriptive features. The SVMs treat each frame of a syllable as a new sample. This is compared to an approach using MFCC features and a nearest neighbour approach using Mahalanobis divergence as the distance metric [6].

In the experiment, the MFCC representations in combination with the SVM classifier performed better on average than the descriptive parameters and reference implementation on the first dataset. The approach using a mixture of all features obtained the best accuracy. On the second dataset, all classifiers performed comparably, with the mixture approach performing slightly better.

3.2.6 Briggs et al.

(perhaps not too relevant)

In their 2009 paper, Briggs et al. [3] compare the accuracy of a bayes risk-minimizing classifier, which is similar to a nearest-neighbour classifier, to SVMs in identifying the vocalisations of six bird species. They do this using a number of different features, including MFCCs, mean frequency and bandwidth and averages.

Spectral density is another feature that is used in the experiments, where spectrum density normalised magnitude spectrum for every frame. This is calculated for every frame as: $f(i) = \frac{|c_i|}{\sum |c_j|}$, where f(i) is the i$^{\text{th}}$ element in a frame, $|c_i|$ is the magnitude of that frame, and $\sum |c_j|$ is the sum of the magnitude of all elements in the frame. Other features that are used are the mean frequency and bandwidth of frames, and MFCCs.

Another feature representation that is tried is is to aggregate the frame-level features into a syllable representation, where low-dimensional feature vectors can be aggregated into histograms. However, for higher-dimensional feature vectors

this would be computationally too expensive as the feature space would increase exponentially with the number of dimensions. Here, instead codebook histograms are used, where every frame is compared to a set of representational feature vectors and the number of frames associated to each representational vector is counted.

Briggs et al. derive a bayes risk minimizing classifier from their Interval-IID (independent and identically distributed) model, and show that it is very similar to a nearest-neighbour classifier using the Kullback-Leibler divergence. Treating the histograms of the syllable representations as discrete probability distributions P and Q with n bins, the Kullback-Leibler divergence is defined as $\sum_i P(i) log \frac{P(i)}{Q(i)}$, where P would be the test histogram and Q the reference histogram. The use of this metric is also compared to the euclidean distance, the manhattan distance and the Hellinger divergence [3]. The last algorithm used in the experiments SVM.

In the experiments, recordings from the Cornell Macaulay library are use, of 6 bird species. The recordings vary in length between 30 seconds and 10 minutes, but are all divided into intervals of 30 seconds, resulting in 413 intervals. Some fragments contain a lot of noise, but the sections with human voices are manually removed. Some intervals contain sounds from multiple species, but they are labeled with the loudest species in the recording. All recordings are made at 44.1 kHz. The recordings are divided into frames of 256 samples, with 50% overlap, on which FFT's are computed. The lowest 8 and highest 64 elements of the FFT are discarded to reduce noise and processing time later on, leaving only 56 elements.

MFCCs are constructed using 24 filters, of which only the first 12 are used in the DCT for the frame-level feature. For constructing the 2D histograms, the values for mean frequency and bandwidth are divided into square bins of 100Hz wide, resulting in 100 bins on the frequency axis and 50 bins on the bandwidth axis, resulting in 5000 bins [3].

For clustering of the codebooks for higher-dimensional feature vectors, the K-means++ algorithm is used, which initialises the centers of clusters, after which norml K-means clustering is done. The clusters are initialised by first choosing a single center at random from the data points, and then iteratively doing the following until k centers are chosen:

- compute the distances $D(x)$ of all data points to their nearest center
- add a center at a data point, which is chosen at random, where the probability of a data point for being chosen is proportional to $D(x)^2$

In their experiments, average representations of mean frequency and bandwidth and of MFCCs are compared using nearest neighbour classification using euclidean and manhattan distance as well as SVM. 2D histograms of mean frequency and bandwidth are used in comparing the Interval-IID approach with nearest neighbout using all four distance metrics. Lastly, codebook representations of MFCCs and Spectral Density features are used in comparing nearest neighbour using all four distance metricsand SVM.

The classifiers are all trained on the data from recordings other than the recording that the test fragment comes from. To reduce random variation in the results using the codebook technique, 5 versions of all these classifiers are created, each using different seeds, and their mean and average deviation is reported.

The experiments show, that regardless of the type of features used, histograms achieve better accuracy than average features. Also, the Interval-IID classifier obtained identical results to a nearest-neighbour classifier with KL-divergence [3]. Furthermore, the Kullback-Leibler distance and Hellinger distances outperformed euclidean and manhattan distances in all cases, and also performed similar to SVMs. Overall, the most effective technique uses Spectrum density as frame features, aggregates these in a codebook and uses a nearest neighbour classifier with the Hellinger divergence.

3.2.7 Brown et al.

Brown et al. compared the accuracy of DTW, GMMs and SVMs as well as the SongScope software[1] (see also Section 3.3) on recognising 608 samples in total of 5 different species. The data was comprised mostly of amateur recordings, which were freely available on the internet, which had been manually cleaned up by removing excessive noise, long periods of silence and noises caused by humans [4].

This data was split halfway into evenly divided training and testing sets. Feature extraction was carried out on both sets. Only for the DTW classifier were syllables extracted, the other classifiers were trained and tested on the whole songs. The features that were used were calculated by dividing the data intor frames or 256 samples, then calculating a FFT and computing the Direct Cosine Transform on those features. Then the data that was lower than one tenth the maximum energy was removed, to be able to ignore insignificant data.

For DTW, the songs were divided into syllables, by clustering the features using k-means clustering and the amount of clusters were decided by maximising the inter cluster distance while minimizing the intracluster distance. The number of mixtures used in the GMM approach was 5, as this was experimentally found to be the optimal value. The data that was used on SongScope was annotated within the songscope environment.

The results of the experiment were, that overall, the GMM approach attained the highest accuracy at 55.2%, followed by the SongScope software at 32%. DTW performed worst with 10%, and the SVM had an accuracy of 23.8%. THe SVM approach, while it had meager results, also took the longest to train and test on.

Sadly, no experiments were done to test the accuracy of GMMs and SVMs at syllable level, even though this was done for the DTW approach and, internally, by SongScope.

3.3 Commercial Implementation (SongScope)

The SongScope software is one of the few commercial implementations available that claim to be able to do bird recognition from their vocalisations. In a set of 2 papers which are available online, the algorithm that is used is detailed, along with specific parameters. The results of experiments on their own database of several species are also published here, but as only the SongScope software is tested, it is hard to compare its performance to other algorithms. The method employed by

[1] By wildlife acoustics, more information available here: http://www.wildlifeacoustics.com/

songscope is, in general, an HMM approach, with the feature vectors comprised of features that are similar to MFCCs, but which are slightly adapted for use with birdsong.

First, an FFT is applied. In the tests, a sample rate of 16 kHz was used, an fft window size of 256 samples and overlap of 50%. The next step is to reduce noise by first applying a Wiener filter to the signal, to reduce stationary background noise [2]. The estimate of the noise spectrum necessary for this filter is calculated by using a one-second rolling average of the spectrum immediately preceding each FFT window. After this a band-pass filter is applied that passes only the frequencies a target species is likely to produce. In the tests, the filter cut everything off outside the range of 1 kHz - 8 kHz.

After this, the frequency domain is warped into a log-scale, which is based on the assumption that higher frequency components are redundant harmonics of the fundamental frequency [2]. The log scale is defined by the following function:

$$F_n = f_a 2^{kn}$$

where f_a is the band-pass filter's minimum frequency, and k is chosen such that the number of log frequency bins is equal to the original number of frequency bins [2] Rectangular filter banks were used instead of the triangular filter banks normally used in MFCCs, because this resulted in greater accuracy.

As the last step, the log power levels of the signal are normalised to a fixed dynamic range. This is done such, that for every FFT window, the frequency bin with the highest energy is set to be the F_{max}. Any bin falling below $F_{max} - dynamic\_range$ or below the noise estimate is set to 0. Typically, the dynamic range is set to around 15-20dB.

Features are extracted by applying a DCT-II filter to the resulting signal. To these features, a relative power level feature is added, and subsequent features which are very similar are represented as one feature vector, with an added length feature. The resulting feature vector size used in the tests is 12.

3.3.1 Syllable Extraction and Model

Syllables are extracted by monitoring the total energy passing through the bandpass filter. For this a number of parameters are tuned for every individual vocalisation [2]: the longest expected duration af any one syllable (Max_{SyD}), the longest expected gap between any two syllables (Max_{SyGD}), the maximum expected duration of a song (Max_{SoD}) and the normalized dynamic range (DR), as discussed earlier. In the tests, Max_{SyD} is set at 1.5 seconds, Max_{SyGD} at 0.5 seconds, Max_{SoD} at 3 seconds and DR at 15 dB.

The energy in a rolling window of length $2Max_{SyD} + Max_{SyGD}$ is monitored to track local minimum and maximum energy values [2]. The onset threshold is normally set at 18dB above the local minimum and the offset threshold at 6 dB below the onset threshold. A syllable begins when the the energy level is above the onset threshold and ends when it falls below the offset threshold. If the gap between two syllables exceeds Max_{SyGD}, a song is completed. This also happens when the song length exceeds Max_{SoD}.

If the vocalisation is more energetic than normal, the onset and offset thresholds are adjusted to be the local maximum energy, minus $DR + 3dB$ for the offset, and 6dB higher than that for the onset. If this happens, any softer vocalisation that was analysed will be ended. Unfortunately, what constitutes more energetic than normal is not specified.

After segmentation into syllables, syllables are clustered into classes by first taking a two-dimensional DCT of the input data and then using k-means clustering. An HMM is then made for every class, with a number of states proportional to the duration of the syllables in the class, with some states representing inter-syllable gaps. The transition likelihoods of different syllable are calculated from the training data, and the individual syllables are trained by iteratively using Viterbi algorithm to segment the data and then training the GMMs [2].

Including large HMMs means that a short vocalisation might be matched by a few states in such a large HMM, thus producing a wrong classification. To counter this, a good match is not only determined by HMM probability, but is also a function of the size of both the input and the used HMM. In the tests, the maximum number of states used by an HMM model is 48, with 1 mixture per state.

3.3.2 Results

The classifier was trained on 12,653 vocalisations, from 550 different recordings of 52 species. The 550 recordings were split into 266 training recordings and 284 testing recordings. Every vocalization is represented on average by 5 training and 5 testing recordings with 23 vocalizations on average in each recording. All classifiers were trained with the same parameters, to be able to run the classifiers in parallel on the dataset, but with tuning, they might have performed better.

The classifiers achieved an accuracy of 63% on the training data, with a false positive rate of 0.3%. On the testing data, the accuracy fell to 37%, and the false positive rate was 0.4%.

The same classifier is used in another paper on SongScope, this time to detect only cerulean warblers [1]. For this some parameters were altered: the sample rate that was used was 20 kHz and the filter cut-offs were 2.8 kHz and 7.5 kHz. Max_{SoD} was set at 3 seconds, Max_{SyGD} at 0.2 seconds, and the size of the feature vector was set at 4.

First this model was trained on 137 manually identified vocalisations from 11 recordings. This was then run on 251 hours of field recordings, yielding 9,145 candidate vocalizations with a score of at least 50% and a quality measure that was high enough. These candidates were inspected and 522 of them were confirmed Cerulean warblers. Due to the high false positive rate, a model was trained again, this time on 61 vocalisations extra, that were taken from the first run. This second model detectd 4,328 candidate vocalisations, of which 1,552 were confirmed, thus decreasing the false positive rate dramatically [1].

3.4 Summary

In the past chapter, several different approaches were discussed, and a literature review was performed. The reviewed scientific literature focussed mainly on techniques that did not incorporate temporal information, in contrast to the commercial implementation that was reviewed. Most studies found that techniques that make use of temporal information (HMMs, DTW) do not perform better, or sometimes even worse than techniques that do not make use of such information. This is in direct contrast to the best performing techniques for human speech at this moment. Intuitively, too it would seem obvious that the temporal characteristics of bird song is useful for recognizing bird song.

Songs as a whole might not have a fixed order, as considerable variation is observed among different members of the same species. However, when observed at the syllable level, any one type of syllable seems to have a relatively fixed order. Thus, it would seem useful to model not only the different points in dimensional space a frame can take, but also the sequence in which those points tend to follow each other.

This approach is also used by the commercial software SongScope, indicating that this might indeed be a better approach. There are several possible reasons why techniques incorporating temporal information have gained less attention in the literature. One is, that for bird song, temporal information really is less important than it is for human speech. Another is, that the temporal techniques that are used, were implemented in a way that made them less well suited to bird song, or the temporal techniques that were used were not as useful as those commonly used for speech.

The current research will continue the line of research that was started by Brown et al. [4], by comparing GMMs, which were the best scoring models that were found by Brown et al. to HMMs, which do incorporate temporal information. The HMMs will make use of gmms for their output distribution. Certain aspects of the SongScope software will be implemented, to make it easier to extend conclusions to the SongScope platform. The same source data as is used by Brown et al. will be used. Firstly the data that is used is discussed, along with the data preparation techniques, including the method used for extracting syllables. The tools which are used to achieved this are briefly discussed as well. After this, the performed experiments are detailed and the outcomes are discussed. Lastly, the results are evaluated and possible lines of future research are explored.

4 Data and Methods

The data is the same as that used by Brown et al. [4]. It consists of recordings of five different species of birds, taken from the internet. The bird species are:

1. Bananaquit
2. Ferruginous Pygmy-Owl
3. Golden-crowned Warbler

Species	Bananaquit	Ferruginous Pygmy-Owl	Golden-crowned Warbler	Road-side Hawk	Striped Cuckoo
Amount of Recordings	115	61	74	160	154
Mean Recording length (in seconds)	1.88	6.99	1.93	2.48	1.70

Table 4.1: Amount of recordings and average recording length for each of the bird species.

4. Roadside Hawk

5. Striped Cuckoo

The amount of recordings for each species, and the mean recording length per species is listed in Table 4.1. Features are extracted from the recordings in a manner similar to that used by SongScope (see section). First off a Wiener filter is applied to the recording, where the background noise is estimated on the least energetic 0.25 seconds. These least energetic parts are identified by applying a Fast Fourier Transform (FFT) with non-overlapping windows to the recording, then finding the time bins, where the sum of the values for all frequency bins at that time are the lowest. The parts of the original recording corresponding to these time bins in the FFT are then appended and used to estimate the parameters of the Wiener Filter.

After applying the wiener filter to a recording, an FFT is applied to the recording with a step size of 8 milliseconds and a window size of 16 milliseconds (corresponding to the 256 frames and 50% overlap that's used in songscope). The window that is used is a Hamming window. From this, the power spectrum is obtained, and a bandpass filter is applied to the data, throwing away all frequency bins under 900 Hz and all those above 10 kHz. These values have been obtained by manually looking at the data and finding the smallest band in which all syllables lay.

A filterbank similar to Mel-frequency filterbanks, but with rectangular filters such as are used in the SongScope software, is applied. The function to define the boundaries of individual filters in the filterbank is described in the SongScope paper [2] as:

$$F_n = f_a 2^{kn}$$

In order to come up with as many frequency bins as in the original FFT (the amount of frequency bins within the bandpass filter), n in this equation is chosen as one more than the original amount of frequency bins, and the frequencies given by this equation are taken as the boundaries of the filters. Frequency bins might remain empty if the boundaries are strictly observed, due to the fact that the filters at lower frequencies are smaller than the original frequency bins. To prevent this,

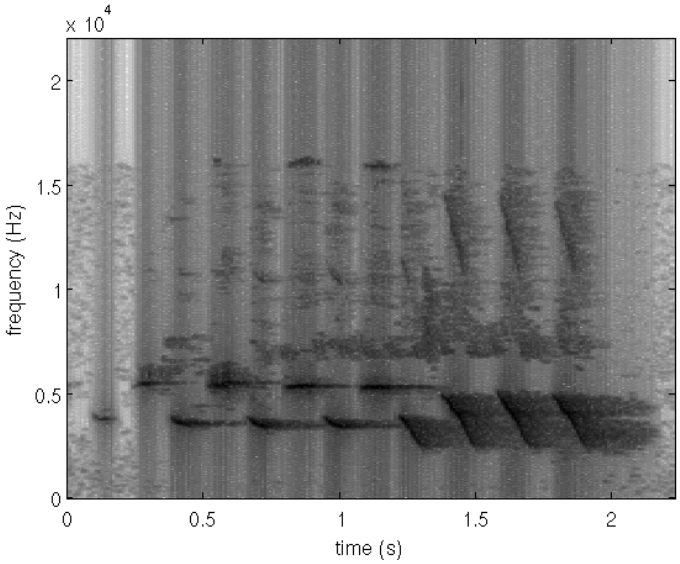


Figure 4.1: Spectrogram of a recording of the Golden-crowned Warbler, darker means more energy. The original recording is already somewhat cleaned up, but some noise remains. Notice that the most important parts of the song are in the lower frequencies and have a relatively high amount of noise trailing.

the frequency bins that each filter in the filterbank covers are the frequency bins closest to the boundaries, and all those in between.

The resulting spectrum is then normalized, by converting the energies to Decibel, and setting the highest energy as 0 dB. Any time-frequency bins which subsequently fall below -20 dB are set to -20 dB, thus removing a large amount of the background noise. This normalization is performed on a per-recording basis, so the maximum energy is found separately for each recording. Lastly, a Direct Cosine Transform (DCT) is applied to the features. Using the above-mentioned bandpass filter, each time-frame will consist of 212 cepstral coefficients.

The combined effect of the filtering, transformation to log-frequency domain and normalization can be seen by comparing Figure 4.1 with Figure 4.2(A). The first of these shows an FFT of the original recording, useing the same settings as is used for extracting the features. The second shows the result after filtering and normalization. Clearly visible is the removal of the noise, but when looking closer, some of the syllable information in the higher frequencies is removed as well. However, this information seems to be an elongated duplicate of the information in the lower frequencies. Also, it is barely more energetic than the background noise, and would have been removed by the normalization if it wouldn't have been removed by the band-pass filter. When compared to the figures shown in chapter 2.2 of the paper for SongScope [2], the result of the noise removal is very similar.

Individual syllables to be used for training and recognition are extracted from the recordings using a similar method as that used by Fagerlund et al. [6]. The base data used for extracting the syllables is the normalized features, before the DCT is applied. The data is first converted back into the power spectrum, then for every time bin, the average is taken for all frequency bins at that time point the

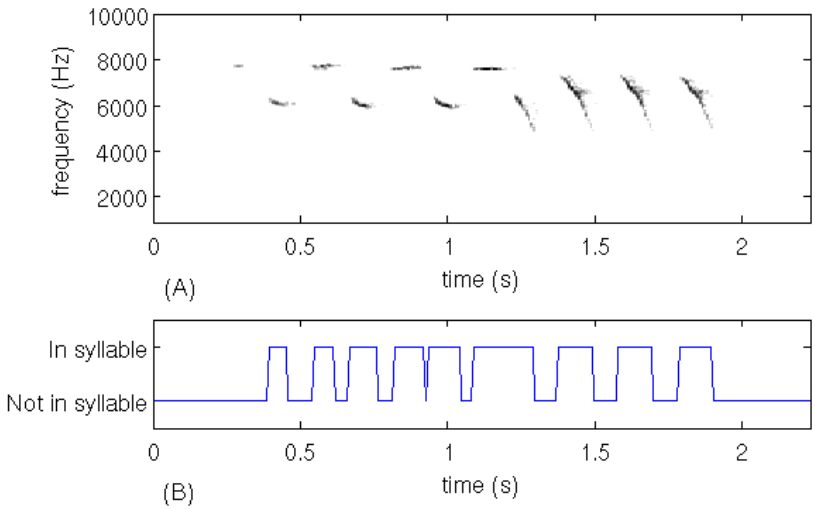


Figure 4.2: Top: Cleaned up and normalized version of recording shown in Figure 4.1, darker means more energy. Notice only the syllables themselves are left and the spectral range is diminished.
Bottom: timeline showing which parts of the recording are included in a syllable. Contiguous parts form single syllables.

resulting vector is smoothed. The smoothing is done by convoluting the vector with with a vector of ones, with length 9. Due to the convoluting process, some extra points are created at the beginning and end of the vector, but these are discarded.

Of the remaining vector, again the first and last 4 values are taken and replaced with 0.01. This is done to remove the steep slope that the resulting vector would otherwise have at these points, which would interfere with the syllable extraction process. The value 0.01 is obtained as the value at which the slope normally ends, due to the fact that the values at the beginning and end of recording are usually only noise, and thus originally 20 dB (which, converted back to the power spectrum, becomes 0.01).

At this point, the values of the vector are increased such, that the maximum becomes equal to 0, and the noise $Noise$ is estimated as the minimum value. A threshold is set at $Factor * Noise$, where Factor is in this case set at 0.9, a value that's obtained experimentally, to maximize the amount of a syllable that's included, while still separating syllables properly. The algorithm then iteratively re-estimates the noise as all bins that fall below the threshold, and the threshold is updated accordingly. This continues until the algorithm converges, after which contiguous parts that are higher than the threshold and longer than 2 frames are counted as syllables.

In figure 4.2(B), the result of using this algorithm on the example recording is shown. Most syllables are correctly extracted, but it can be seen that the first, smallest syllable is dropped, and in the middle, two syllables are extracted as

Species	Bananaquit	Ferruginous Pygmy-Owl	Golden-crowned Warbler	Road-side Hawk	Striped Cuckoo
Amount of Syllables	493	971	363	579	395

Table 4.2: Amount of syllables for each of the bird species.

though they were one. The algorithm works well enough to be used however, and the amount of syllables extracted per species is shown in Table 4.2.

4.1 Tools

The feature extraction and syllable extraction was done in Matlab. For all the experiments, the code to distribute the syllables to different groups was also written in Matlab. For the simpler GMM experiments, the code was written in Matlab, making use of the NetLab Toolkit, which provides routines for making several statistical models in Matlab. It also provides an implementation of the k-means clustering algorithm that's used for some of the experiments. For the experiments involving HMMs, the Hidden Markov Toolkit (HTK) is used, which is:

> (...) a portable toolkit for building and manipulating hidden Markov models. HTK is primarily used for speech recognition research although it has been used for numerous other applications (...) [2]

In the case of GMMs, initialization was done by pooling the training data for a GMM, and initializing a 1-mixture model on the means and covariances of the training data. To obtain GMMs with more mixtures, the mixture with the largest prior weight is split in two, with each mixture being altered by a proportion of the original variation, in different directions. Expectation-Maximization is then performed six times, to train the model. These last two steps are repeated as often as is necessary to obtain the desired amount of mixtures.

Recognition is then performed by obtaining the average logprobability for a sample, given a model, and the model which gives the highest probability is the model that is recognised.

In the case of HMMs the HInit tool is used for initialization, which initially segmentats training files uniformly, meaning that, each state will get n/m feature vectors assigned to it, where n is the amount of feature vectors in the training file, and m is the amount of states. As only left-to-right models are used to model syllables, the first state gets assigned the first n/m feature vectors, the second state the second group, etc.

The GMM model that is associated with a state is then initialized, by clustering the data that is associated to the state using k-means clustering, and using the resulting centres as the centres for the mixtures. The covariance of the data associated with each centre is then used as the covariance of that mixture, and finally

[2] `http://htk.eng.cam.ac.uk/`

the proportion of feature vectors associated with a mixture relative to the amount of feature vectors associated with a state is used to initialize the prior probability of that centre.

Expectation-Maximization training is then performed by segmenting the training files using the Viterbi algorithm, and updating the GMM models associated with a state as above. This is done until the model has converged, or until 20 passes of EM are performed, whichever comes first.

Recognition is performed using the HVite tool, which does viterbi recognition using the token passing algorithm. The HMM network that is used depends on the grammar that is used. In the simplest case, recognition on single syllables with, one model per species, the network consists of all the bird species models in parallel, meaning only one of those models can be used for the entire recording.

5 Experiments

Several different experiments are performed on the data, to determine the effects that different amounts of temporal information have on the accuracy of the recogniser. Firstly, syllable-level recognition is performed using GMMs, to determine the optimal combination of number of mixtures and features. In this experiment, for all syllables of each species, one model will be created. The feature extraction process results in 212 features per frame, which is vastly more than the number normally used for MFCCs in human speech recognition (which is usually around 12). Because the features used in these experiments are analogous to MFCCs, it is expected that a reduction in syllables is appropriate.

After this, an experiment is performed to compare the performance of HMMs with different number of states and mixtures, to see if an increase in accuracy can be obtained by modelling temporal information within a syllable. It is expected that such a HMM using only one state will perform similarly to a GMM with the same amount of mixtures.

In the following set of experiments, the syllables are clustered into groups, and a single HMM is used to model each cluster, instead of modelling all syllables of a species using only one HMM. Firstly, the optimal number of clusters for each species is determined for use with a GMM, modelled as a HMM with one state, after which it is explored whether more accuracy can be obtained by increasing the amount of states. Lastly, it is explored how much can be gained by performing recognition on entire recordings, using either GMMs or HMMs, and by using clusters in this context.

In all experiments, 10-fold cross-validation is used, by dividing the data for each species into 10 groups of nearly equal size. Syllables from the same recording can be expected to be similar, because birds often make the same syllable a number of times in a row. Similarly, recordings are read in sequentially, and any two consecutive recordings might feature the same bird. To offset any bias because of this, for each experiment run, the samples are divided semi-randomly into one of the groups. This is achieved by creating a random permutation of the samples, and then dividing it into 10 consecutive groups.If the data couldn't be divided exactly

into 10 groups, the remaining data is assigned to the last group.

For each of the groups, a model is then trained on the other groups, after which recognition is performed on the withheld group. This results in 10 values for accuracy for each of the ten groups. The mean of this value is then taken as the estimated true accuracy. This was done, because earlier tests had shown that when models are trained on half of the data, randomly selected from the total pool of data and then tested on the other half, the accuracy could differ by as much as a few percentage points, in absolute terms, not relative to the original accuracy.

In order to determine whether two different values of mean accuracy are really different, the two-sample t-test will be used, which compares two vectors of random samples, in this case, the two vectors of 10 values for accuracy. There is no a priori expectation for either of the samples to be larger, as any change in parameters might either facilitate better recognition, or merely increase complexity and cause more overfitting to occur. Thus, the two-sided test will be used in all cases. For the same reason, variance cannot be assumed to be equal between two samples: it might be that one sample is well trained and thus has relatively small variance, whereas the other either gives very high or very low accuracy due to overfitting (in the cases where the test data is very different or very similar to the training data). Thus, unknown and unequal variance is assumed, and Satterthwaite's approximation for effective degrees of freedom is used.

In order to be able to do these tests, the assumption is made, that these values are normally distributed. This doesn't have to be the case, but is a reasonable assumption, given that the training and testing samples are randomly distributed to each of the 10 groups. The value for accuracy of each group is thus an approximation of the true accuracy, from a random distribution which should have true accuracy as mean. The significance level will be set a 0.05 for all tests that will be done.

5.1 Recognising with one model per species

Firstly, it is of interest to determine how many cepstral coefficients should be used in a frame. In MFCC's usually the first 12 or so coefficients are used, from the approximately 20 that are computed. However, in this case, the amount and type of filters used to compute the cepstral coefficients is 212, many more than are computed in MFCCs. Thus, it is unclear how many cepstral coefficients are needed to facilitate recognition without unnecessarily increasing the complexity of the models. This number could be 13 as well, but perhaps the same proportion of coefficients that are computed should be used.

Thus, in the first experiment, GMMs are trained with 1 through 6 mixtures, using the first n features (cepstral coefficients) per frame, where n is either 10, 20, 30, 40, 50 or 100. The condition with 100 coefficients is included as an extreme option, in order to be able to interpret the results relative to an extreme.

In the second experiment, HMMs will be used to perform recognition, with from 1 to 5 mixtures and also from 1 to 5 states. The amount of features that will be used is determined by the outcome of the first experiment. The outcomes with 1 state will serve as a baseline for this experiment, as they are nearly equivalent to

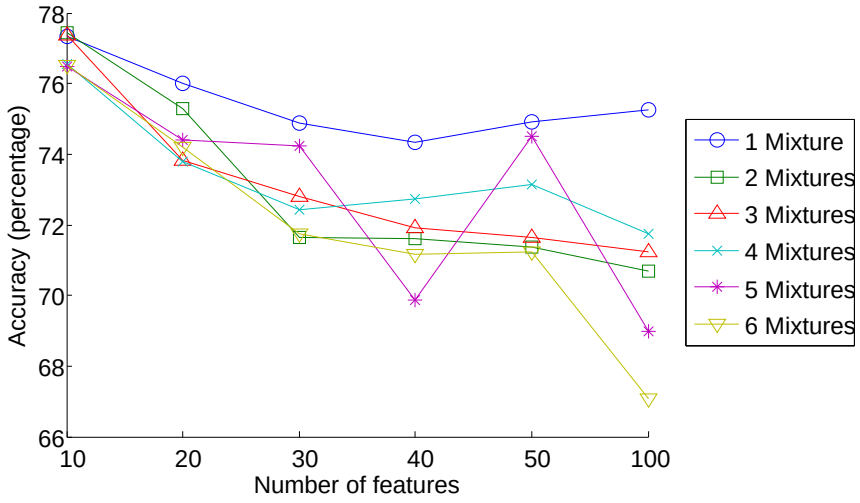


Figure 5.1: Accuracy of GMMs on recognising syllables, plotted against the amount of features used. Each line refers to a different amount of mixtures used.

using GMMs with the same amount of mixtures.

5.1.1 Varying size of feature vectors and amount of mixtures in GMMs

The results are summarised in Figure 5.1, and seem to show that, regardless of the amount of mixtures that are used, when the features start to become higher than 10, the accuracy drops. However, most of these differences, with a few exceptions, are in the range of 2 points of accuracy difference between nearby data points. This is problematic, as the standard deviation for all the accuracy values ranges from 2 to about 5, with a few outliers that are higher.

In other words, these differences might be due to random variation, perhaps due to insufficient data. To investigate this possibility, statistical tests can be used on some of the more extreme examples of differences in parameters between models. Firstly, to determine the impact of amount of features used, the results on using 10 features can be compared with those of using 20, 30 and so on until 100. This would give an idea of roughly how many features is definitely too much and bad for the accuracy. It would be expected that this figure is different when a different amount of mixtures is used, since more mixtures means higher complexity, which is multiplied by the higher complexity caused by using more features. Thus, with more mixtures, an overfitting effect could be expected to occur sooner with less features than otherwise.

To test for this, the two instances where the difference between that instance and using 10 cepstral coefficients are compared for every amount of mixtures. It is assumed that, if the lowest values do are not significantly different, values closer together will also not be significant. The results are summarised in Table 5.1. For most of the values, the results are significant, except for most of the cases with 5 and 6 mixtures.

In the figure, it seems the accuracy only increases with less features. However, at some point, this should drop again, as there would not be enough information in

Amount of mixtures in second model	Largest difference	Second largest difference
1	[40]: significant, $t(15.1) = 2.43$, $p < 0.05$, $CI = (0.37 - 3.64)$	[30]: significant, $t(16.8) = 2.29$, $p < 0.05$, $CI = (0.19 - 4.73)$
2	[100]: significant, $t(12.6) = 3.55$, $p < 0.05$, $CI = (2.63 - 10.9)$	[50]: significant, $t(13.3) = 3.49$, $p < 0.05$, $CI = (2.32 - 9.8)$
3	[100]: significant, $t(15.4) = 3.53$, $p < 0.05$, $CI = (2.42 - 9.81)$	[50]: significant, $t(13.9) = 2.89$, $p < 0.05$, $CI = (1.46 - 9.97)$
4	[100]: significant, $t(17.1) = 2.5$, $p < 0.05$, $CI = (0.77 - 8.86)$	[30]: significant, $t(17.57) = 2.25$, $p < 0.05$, $CI = (0.27 - 8.01)$
5	[100]: insignificant, $t(11.2) = 1.97$, $p > 0.05$	[40]: insignificant, $t(10.65) = 1.52$, $p > 0.05$,
6	[100]: significant, $t(11.5) = 2.43$, $p < 0.05$, $CI = (0.91 - 17.9)$	[40]: insignificant, $t(11.7) = 1.43$, $p > 0.05$,

Table 5.1: T-test statistics, the amount of features for the second sample is in square brackets. CI denotes the confidence interval for the difference of the two samples. All tests are compared to results on the model with the same amount of Mixtures, but with 10 cepstral coefficients.

the features to classify on. Therefore, another experiment is carried out, ranging the amount of features from 2 to 15, with a 1-mixture GMM model. This is done, to make sure there is enough information in the data so even the simplest model still attains reasonable results, and the results reflect the performance of different models, not a lack of information in the data.

The results are shown in Figure 5.2. The accuracy only starts to noticeable drop when the amount of features drops below 4. For the rest of the results, the are very near to each other, and don't show a general trend in either direction.

5.1.2 Evaluation

The results indicate that, at least when using 4 or less mixtures, it is better for performance to use less than thirty features. Possibly, this holds for more complex models as well (such as those using more mixtures), but the results had too much variation to be certain. It is also possible, that for these cases, there really was no difference, but it is impossible to be certain with these results. Quite likely, even with the least complex model, no difference can be found between using 10 or 20 features, due to the variance in the results. Subsequent testing finds this to be true ($t(17.9) = 1.38$, $p > 0.05$), but again, this may be due to variance and data

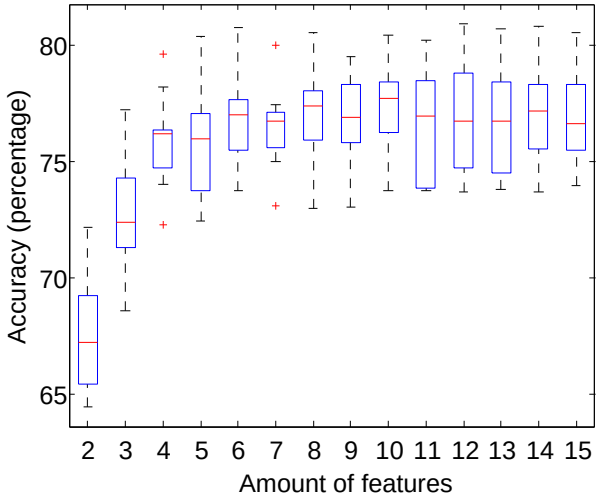


Figure 5.2: Boxplot of the accuracy of a single mixture GMM on data with varying amounts of cepstral coefficients, ranging from 2 to 15. The boxes denote the 25^{th} and 75^{th} percentile, the bar in the middle the median, the whiskers the maximum range. Pluses denote outliers.

scarcity, rather than that there is no actual difference.

Furthermore, the results show that the amount of features that are used does not seem to have a large impact on the accuracy, as long as there are at least 4 cepstral coefficients per feature vector. However, the amount of cepstral coefficients that is used should not be too high, as there is then a heightened chance of overfitting. In the experiments that are described in the following sections, 12 cepstral coefficients will be used in the feature vectors, in all experiments. This gives enough features to allow models to extract enough information, but likely not so much that the models will show overfitting too easily.

5.1.3 Varying amount of states and mixtures for HMMs

These results show that, on the whole there is a tendency for accuracy to go up, when the amount of states is increased. The results on the whole also show higher accuracy than was obtained with the GMM models, even when using only one state. The differences between the results in this graph also are lower, than in the first experiment in this section, which might make it harder to show a significant effect, if the variance is similar.

Looking at the difference in results when using more states, an interesting question is whether the results between using 1 and 5 states is significant. For the case where 1 mixture is used, this is not the case but, for all other cases, the difference is significant (see Table 5.2 for the test results).

Similarly, the effect of adding mixtures while keeping the amount of states equal can be investigated. This data is summarised in table 5.3. Here, too, for most of the tests, the results were statistically significant. The results on the models with 1 state provide a counterexample though, just as the models with 5 states.

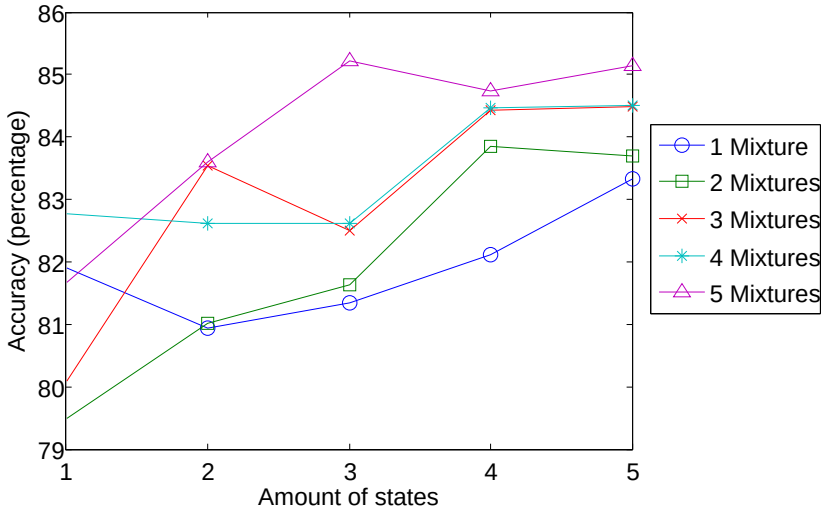


Figure 5.3: Accuracy of HMMs on recognising syllables, plotted against the amount of states used. Each line refers to a different amount of mixtures used.

Amount of mixtures	Test result
1	insignificant, $t(17.5) = -1.65$, $p > 0.05$,
2	significant, $t(13.4) = -5.36$, $p < 0.05$, $CI = (-5.88 - -2.51)$
3	significant, $t(15.2) = -5.78$, $p < 0.05$, $CI = (-6.03 - -2.78)$
4	significant, $t(17.9) = -2.48$, $p < 0.05$, $CI = (-3.23 - -0.26)$
5	significant, $t(17.6) = -3.91$, $p < 0.05$, $CI = (-5.34 - -1.60)$

Table 5.2: T-test statistics comparing results for models with 1 state to those with 5 states. CI denotes the confidence interval for the difference of the two samples.

5.1.4 Evaluation

The results for the second experiment seem to indicate that, at least up to 5 states and 5 mixtures, increasing complexity tends to increase accuracy. However, when using more than 1 mixture, there seems to be no increase in accuracy between using 4 and 5 states. Note, that this seems to happen earlier for the model using 5 mixtures, where accuracy stops increasing after using 3 states. It might be, that this is the result of overfitting, due to increasing complexity, or even random chance.

Note that all the results are higher than those obtained when using pure GMMs. This is also true when using only one state, which should be nearly the same as using GMM models. Most likely, this is due to the difference in techniques to initialize the models (upmixing vs k-means), and the fact that the HMM models

Amount of states	Test result
1	insignificant, $t(17.4) = 0.27$, $p > 0.05$
2	significant, $t(17.9) = -2.32$, $p < 0.05$, $CI = (-5.04 - -0.25)$
3	significant, $t(16.2) = -3.94$, $p < 0.05$, $CI = (-5.95 - -1.79)$
4	significant, $t(17.6) = -2.59$, $p < 0.05$, $CI = (-4.75 - -0.49)$
5	insignificant, $t(17.7) = -2.06$, $p > 0.05$

Table 5.3: T-test statistics comparing results for models with 1 mixture to those with 5 mixtures. CI denotes the confidence interval for the difference of the two samples.

typically have more rounds of E-M training with the correct amount of mixtures.

The data also indicates, that the effect of using more states is slightly more effective than using more mixtures. This can be seen in the tables with the t-test results, for models with the same complexity. As adding one mixture increases complexity just as much as adding one state does, the n^{th} row of table 5.2 and of 5.3 show results of models with the same complexity. The confidence intervals for the 2^{nd} and the 3^{rd} entries have higher values for the tests comparing models with different amounts of states. However, this is not true for the 4^{th} entry.

Also, the 5^{th} entry is only significant in the test involving a change in the amount of states. These differences are relatively small, though, and might have come about due to chance. One reason why adding states is not much better than adding mixtures could be, that in the ultimate calculation of the probability for the model, the transition probabilities have much less influence than the observation probabilities. This is because the observation probability gives n inputs to the calculation of the probability per frame, where n is the amount of cepstral coefficients, while the transition probability only contributes 1 factor to the frame probability. However, this reasoning ignores the fact that in an HMM, the available mixtures per frame is time-dependent, whereas for GMMs, any mixture can contribute to calculating the probability.

5.2 Recognising with multiple models per species

As was shown in the introduction, a single species can use multiple distinct syllables to create a song. The kinds of syllables that are used as well as how many syllables there are, varies per species. It makes sense then, to try to use a distinct model for each of these syllables. As each syllable is modelled by a separate GMM or HMM, the accuracy should go up.

However, it is unknown beforehand, how many syllables there are for each species, and as they are not identified by hand, it is necessary to use a clustering algorithm to do this. To work around this problem, the same approach was taken here, as is used in the SongScope software. Thus, models will be created using a

number of differing amounts of clusters for each species, after which the effect on accuracy is measured. Because each species is expected to have an optimal number of clusters (namely, the amount of differing syllables in the song of that species), the effect of increasing the amount of clusters for each species, while holding that of other species on 1 cluster, is investigated.

In the experiments, in all conditions, the amount of cepstral coefficients that will be used will again be 12, 3 states will be used and 2 mixtures. These values were chosen, as a system that would still perform relatively well, but will not easily fall prey to overfitting. As we expect the models that need to be trained to be slightly simpler in most conditions (ideally exactly 1 syllable per model, instead of multiple), overfitting might occur earlier than in single model per species experiments. In all conditions, the amount of models will be increased for only 1 species, while keeping the amount of models for the other species at one each. This way, it is most likely that the correct amount of models can be found for each species. The amount of models per species will go up to 6.

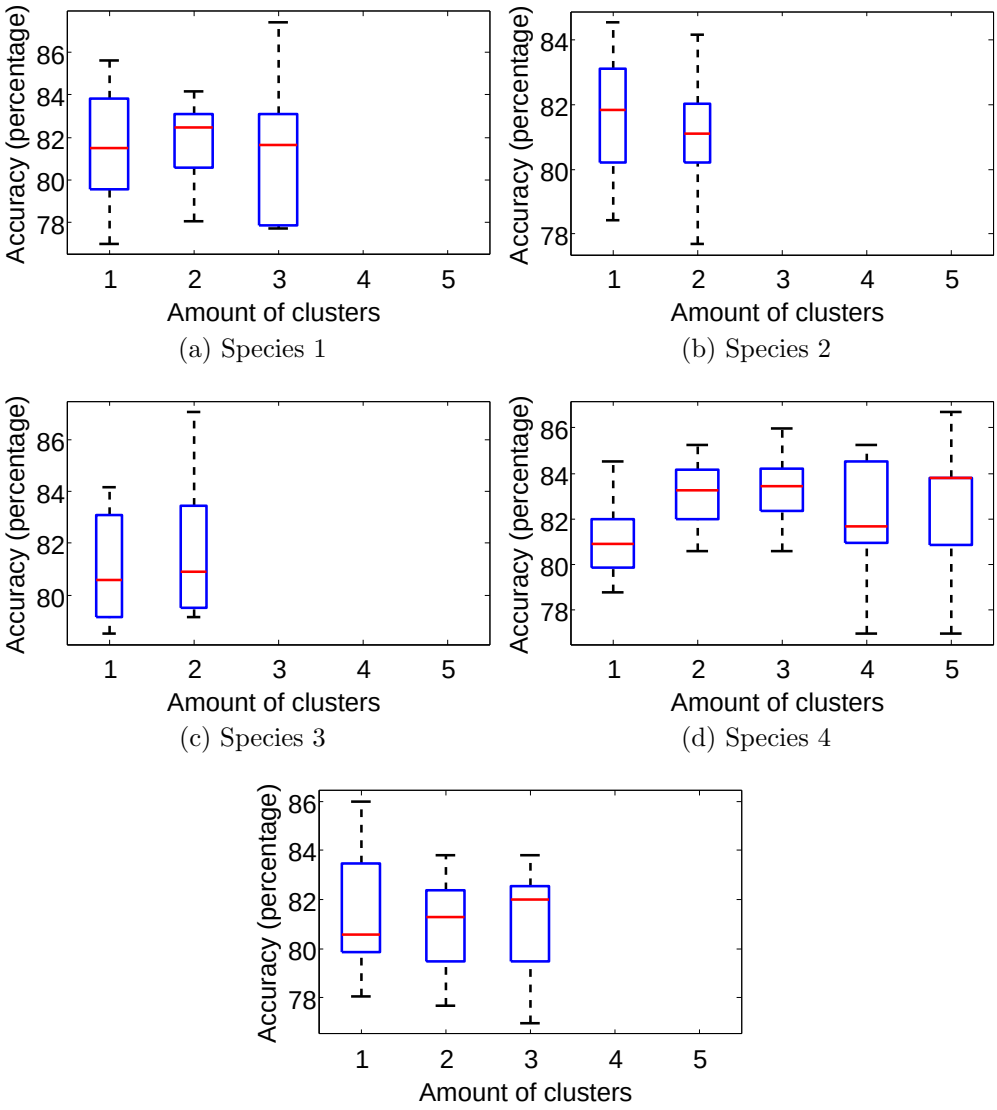
In the case where one of the clusters in the from the k-means algorithm has no samples associated with it, no recognition is performed, as it would be similar to doing recognition with one less center. Most likely, if this happens, the number of syllables for a species is the maximum amount of groups that have samples associated. In the next section, the used algorithm will be detailed, after which the results of the experiments will be reported.

5.2.1 Methods

The k-means clustering algorithm was used to do the clustering. Because the k-means algorithm expects all feature vectors to be of the same length, for each species, the feature vectors were zero-padded to have the same length as the longest syllable for that species. A DCT was then taken over all the time bins for every frequency bin, and the resulting vector was used as input for the k-means clustering algorithm. The k-means clustering algorithm uses the squared euclidean distance as its distance function and is initialized by n random samples datapoints, where n is the number of clusters. This results n centers, which are subsequently used to label the same data again.

A separate model is then trained on each of the clusters separately, which can result in several different labels per species. However, when recognizing on syllables, only one model may be used per syllable, so the label that's recognised can easily be converted back into a more general label denoting only the species that is recognised.

All of the experiments below are performed in HTK, meaning technically a HMM is always used. However, when the individual syllable clusters are modelled with a HMM with one state, the effect will be similar to using GMMs. This is partly due to the fact that no statistical language model is used, and also due to the grammar that is used in the model, which allows only one model to be used in a single recording. To prevent accuracy being calculated such, that it matters which model from the same species is used, the output is cleaned up after recognition, to reflect only which species is recognised, not which syllable model.



(a) Species 1 (b) Species 2

(c) Species 3 (d) Species 4

(e) Species 5

Figure 5.4: Accuracy of HMMs with 3 states and 2 mixtures, and varying amounts of syllable models per species. The amount of syllable models is only changed for one species at a time. The numbers of the species correspond to the numbers in Section 4. The boxes denote the 25^{th} and 75^{th} percentile, the bar in the middle the median, the whiskers the maximum range.

5.2.2 Results

The k-means clustering algorithm had at least one empty group for all species when trying to create 6 clusters. Only for the fourth species (Roadside Hawk) could five clusters be created, the others had maximally 2 or 3 clusters. The results are shown in Figure 5.4. The means for the results are listed in Table 5.4. As can be seen, the means barely change at all, and given the variance shown in Figure 5.4, in most of the cases no significant difference can be expected. However, for the fourth species, there is a somewhat larger difference between using 1 cluster and using 2

Species	1 cluster	2 clusters	3 clusters	4 clusters	5 clusters
1	81.6220	81.8850	81.3810		
2	81.6160	80.9690			
3	81.0180	81.6870			
4	81.2190	83.0660	83.2770	82.1520	82.4750
5	81.3860	81.0510	81.1310		

Table 5.4: Means for the test results where the amount of clusters is changed for each species, one at a time.

to 3 clusters. Two-sample t-tests show that for these conditions, there is indeed a significant difference (for the difference between 1 and 2 clusters, $t(16.5) = -2.42$, $p < 0.05$ confidence interval is $-3.46 - -0.23$ and for the difference between 1 and 3 clusters, $T(17.4) = -2.58$, $p < 0.05$, confidence interval is $-3.73 - -0.38$).

There might be a cumulative effect though, where using multiple clusters for each species yields a measurable increase in accuracy, where doing so for one species at a time doesn't. To test for this, recognition was also performed using 3, 2, 2, 3 and 3 clusters for each species respectively. The mean accuracy did seem to go up to 84.7%, compared to 83.27% when only increasing the amount of clusters to 3 for species 4, however this effect was not statistically significant ($t(17.9) = -1.92$, $p > 0.05$).

Lastly, there might be interaction between the amount of states and mixtures used and the amount of clusters per species. With more clusters, there would be less training material for each of the HMM models, so the optimal number of states and mixtures might change. Thus, another experiment was done, using 2 clusters for each species and varying the amounts of states and mixtures between 1 and 5 each. The results of doing this is shown in figure 5.5. Again, with more states and more mixtures the accuracy tends to increase.

Comparing Figure 5.5 with Figure 5.3, at almost all the points, the accuracy in the former is higher than that in the latter. Furthermore, the maximum accuracy in Figure 5.5 is just over 86%, whereas in Figure 5.3 it is only just over 85%. When comparing the individual datapoints from section 5.1.3 with those from this section, the differences are generally relatively small, all around 1 or 2 percentage points, with only a few exceptions. As the standard deviation for these numbers is also around 2 for all of them, it is very unlikely that the accuracy at those individual points will differ significantly between the cases of using 1 and 2 models to model a species. However, the data do show a strong tendency that using more HMM models is beneficial, as out of all the 25 conditions, the average accuracy is higher in 22 for the cases where multiple models per species are used. Note also, that again using more states brings a similar increase in accuracy as using equivalently more mixtures.

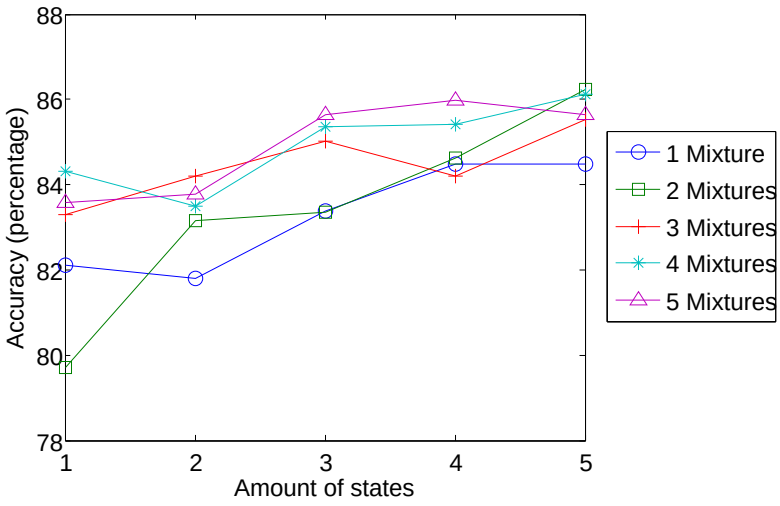


Figure 5.5: Accuracy of HMMs on recognising syllables, plotted against the amount of states used. Each line refers to a different amount of mixtures used. For every species, 2 HMM models were used for recognition.

5.2.3 Evaluation

For most of the species then, training multiple models for that species will most likely not have much of an effect. However, for one species there was a significant effect, and if multiple models are trained for all of the species, the accuracy does seem to go up. In this case, there might be two effects going on: the increase in accuracy when multiple models are trained for all species, might be due only to the increase that was observed for species number four (the Roadside Hawk). Another possibility is, that for the other species, accuracy does increase, but by a small amount and this is not measurable on its own. When doing this for multiple models, the cumulative increase becomes large enough to measure. The current data does not support this last option though, as the mean accuracy of most species either stayed nearly the same or dropped slightly when adding models for only one species at a time.

The results obtained in this experiment were rather unexpected. Using more models for species, preferably one for each class of syllable that can be distinguished, seems likely to increase the accuracy. Most of the species have multiple syllables, as is clear in the spectrogram of the song of the Golden-crowned Warbler, in Figure 4.1. It seems to have 3 distinct syllables, which should be relatively easy to model. The results then might be the effect of insufficient training data. There are a limited number of examples for each species, so the more it is split up into distinct clusters, the less well trained the individual models will become. This would mean that for models trained on larger datasets it should give a clear increase. Another possible explanation might be, that the accuracy for the other species was already so high, that any increase was hard to obtain, but for species 4 it was possible.

Regardless of whether it gives a clear increase for all species, the results indicate that it might be beneficial to do even if it only seems useful for one species. The

results do give a clear tendency to be higher when multiple models are used, but the difference between these results isn't large enough to be certain. When looking at every condition independently, where m mixtures and n states are used, the difference between using 1 and 2 models per species aren't clear. However, the fact that nearly all of the conditions show an increase of average accuracy is a good indication that it is useful to use multiple models.

5.3 Recognising on whole recordings

One last way where recognition performance can be increased, is by using entire recordings to perform recognition, instead of using single syllables. This mode of recognition is also closer to how the technique might actually be applied, as for recognition the data generally hasn't been split into individual syllables. The way this is implemented for the current experiment is, to train the models on individual syllables extracted from the recordings, using the viterbi algorithm as before. The models themselves will be HMMs, and there will be only one model per species, as only the change in accuracy due to using long recordings needs to be recognised.

In order for this to work, a silence model has to be trained as well. It is simply trained on all the data that lies between syllables. The recordings all start and end with a syllable, so there is no silence at the beginning or end of the recording. To make sure only one species is recognised, the HMM network is constructed such, that any one recording may consist of the HMM for one of the five species, optionally followed by one or more sequences of silence followed by a model for the same species. This way, only syllables for one species and silence can be recognised. Again, the output from HTK is cleaned up afterwards to show only which species is recognised, not which syllable model and where silence is recognised. This way, accuracy can be calculated based on the recognised species.

5.3.1 Results

In this case, too the results for every condition is over that obtained in section 5.1.3, with most of the average accuracies being near or over 85%, with one condition even attaining 88.5% (3 mixtures, 5 states). However, the standard deviation for these experiments is also higher, with most being around over 3 (save for 2 conditions). This means that, again on a case by case basis, for every condition in this experiment, there is no certainty that its equivalent in the experiment in section 5.1.3 will be significantly different. The mean results are higher for all conditions in this experiment than when performing recognition on syllables. Again, the increase in accuracy when using more mixtures is similar to the increase obtained by increasing the number of states.

5.3.2 evaluation

The higher mean values obtained in this experiment were most likely due to having more samples for every species to decide on. However, it does show what kind of results might be obtained in a more realistic setting, where multiple syllables from one bird are likely to follow each other. Although no definitive result could

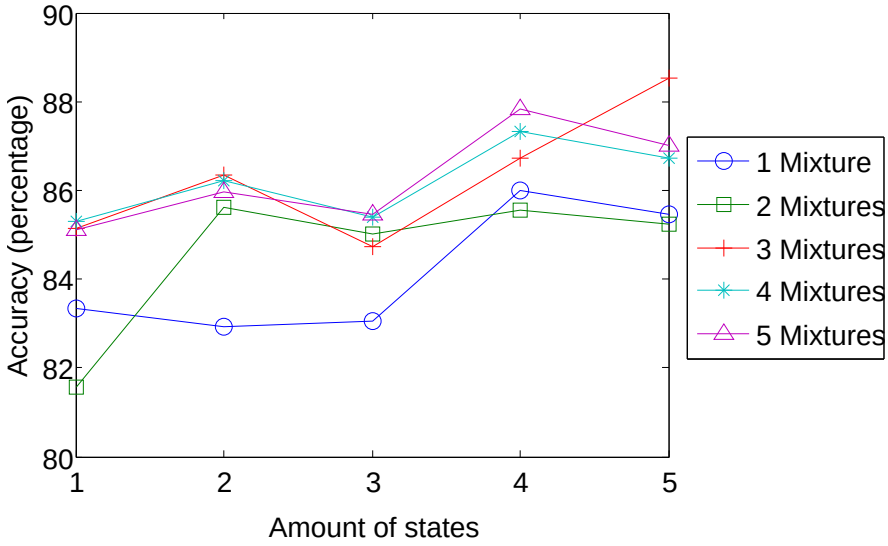


Figure 5.6: Accuracy of HMMs on recognising full recordings, plotted against the amount of states used. Each line refers to a different amount of mixtures used. For every species, 1 HMM model was used for recognition.

be obtained, it does indicate that it is likely that using full recordings increases accuracy by a fair bit.

The high standard deviation observed in the data is most likely due to having few recordings available to do testing on, for one species as low as 6 recordings for testing. This means that each individual estimate for accuracy is likely to vary a large amount, thus causing a high standard deviation. It's very likely that, if more recordings were available, the resulting estimates for accuracy would show less variation.

Interesting is, that the mean accuracy with 3 or more mixtures is very similar, indicating that with more data, the amount of mixtures is less important. It could be that the larger amounts of data mean that it is more likely that a particularly telling piece of data is available, which causes wrong models to give very low probability. This is only speculation however, as this feature of the results could also be an artifact of random variation.

6 Discussion

The experiments that were discussed show in all cases a higher accuracy than was obtained by Brown et al. [4], where, over all the samples together the highest accuracy was obtained with a GMM, at 66.4%. This is lower than even the mean result obtained by a single mixture GMM, on feature vectors consisting of 2 cepstral coefficients in the experiments here. This was obtained by performing recognition on full recordings, not syllables, which was only done for their application of DTW.

Most likely, the difference in results is due to the kind of features that were used. In the experiments by Brown et al., an FFT was applied to the windowed

data, and immediately after that, a DCT was taken of the results of that FFT. This means there was no attempt to remove the noise from the data, which could explain the lower results. Also, the data isn't transformed into a log-frequency domain, nor is the data normalized.

Furthermore, the entire frequency range is used, instead of only that range in which the syllables lay. This may give the system extra data that is of no use and may cause the system to make errors. Lastly, no mention is made of the amount of coefficients that are used. As the experiments in section 5.1.1 show, if they simply used all coefficients (256 in their case), this will have had an adverse effect on the performance.

In one of the papers on Songscope [2], the authors train 54 different vocalization classes for 52 bird species and then test on whole recordings, calculating accuracy as the proportion of vocalizations in all recordings that are correctly recognised. When performing recognition on the training data, the accuracy was 63%, whereas this drops to only 37% on the testing data. This is much lower than the values obtained in the experiments detailed here, which is most likely due to at least two factors. Firstly, many more bird species were used, which makes the baseline accuracy (simply guessing for each vocalization) much lower, and the chance of having two bird species with very similar vocalizations increases.

Secondly, the recognition was performed over long recordings, sometimes with multiple vocalizations and one of the goals was to keep the false positive rate low. In doing so, the true positive rate will inevitably go down as well. Unfortunately, this makes comparison to the results from the current experiments rather hard, and not much more can be concluded.

Comparing with the results from Kogan and Margoliash [9], the results obtained here were lower, however the accuracy in the experiments by Kogan and Margoliash was computed for recognition on each bird separately, with only 2 bird species in total. Also, the training samples were generally included in recognition, which further increases accuracy. Strikingly, accuracy tended to drop very much when performing recognition on entire recordings and songs instead of syllables. This may be due to a lack of a silence model though, as silence models aren't mentioned in the article.

In an experiment by Somervuo et al. [11], using the entire songs of 13 species, both GMMs and HMMs attained a maximum of 70% using MFCCs with delta and delta-delta coefficients. Given the results obtained here, there seems to be some evidence that HMMs could be of use in recognition, which would contradict these results. However, different bird species are used and the evidence here indicates only marginal improvement over simply using more mixtures. Once again, the lower accuracy might be due to there being more models, rather than the techniques employed here.

Similarly, Lee et al. [10] perform recognition on two sets of songs, with 420 and 561 species respectively using codebooks and MFCCs, but obtain a maximum accuracy of 87% and 86% respectively. Since the same accuracy is obtained here with only 5 species, it is tempting to conclude that Codebook MFCCs might be more suited to recognising birdsong. These experiments have only 1 representative song for each species though, and perform recognition on the same samples as

training is performed on. The results can therefore not easily be compared. For the same reason, a comparison with the results by Tyagi et al. [12] can not be made.

Briggs et al. [3], however do train and test on different samples, taken from recordings of 6 different bird species. Using spectrum density codebooks and the nearest neighbour classifier, an accuracy of 92% could be obtained. This accuracy could not be exceeded in any of the experiments described in section 5, and doesn't use any temporal information, and thus seems to perform somewhat better.

7 Conclusions

The performance of GMMs and HMMs in different configurations was compared, in order to establish the influence temporal information can have on the recognition accuracy. Feature extraction methods similar to those used in the SongScope software were employed, resulting in cepstral coefficients, similar to MFCCs. Syllables were extracted from the recordings using an iterative algorithm based on that used by Fagerlund et al. [6]. for some of the experiments, syllables needed to be automatically grouped. This was done using the k-means clustering algorithm.

Experiments were carried out performing recognition on syllables using GMMs and HMMs, with varying amounts of cepstral coefficients, mixtures, states and varying amounts of models per species. The results of varying the amounts of cepstral coefficients suggest that it is not beneficial to use significantly more than 20 cepstral coefficients in a feature vector. For the subsequent experiments, 12 cepstral coefficients were used.

The results of those experiments suggest, that using multiple-state HMMs may give slightly higher accuracy than using GMMs or single-state HMMs, however the difference is small and not statistically significant. Furthermore, the effect of using multiple models to represent different syllable classes for the same species seems to be much larger than the difference between HMMs and GMMs, as is the effect of performing recognition on whole recordings instead of syllables. Overall the results indicate that, at least in the case of HMMs, temporal information will increase accuracy only a slight bit, if at all.

Lastly, the difficulty in comparing the results of different methods in other papers highlights the need for a more standardised dataset and standardised tasks. The high variation in the results obtained in this paper also indicate that a larger dataset is needed to obtain more definitive results. This applies not only to birds, as there are multiple species which might be identified by the sounds they make, and for which it could be beneficial to easily determine if they are present in an area, such as endangered species.

References

[1] Ian Douglas Agranat. Automatic detection of cerulean warblers using autonomous recording units and song scope bioacoustics software. online, 2007. `http://www.wildlifeacoustics.com/songscope/cerulean/AutomaticDetectionOfCeruleanWarblers.pdf`.

[2] Ian Douglas Agranat. Automatically identifying animal species from their vocalisations. online, 2009. `http://www.wildlifeacoustics.com/songscope/aiasftv/aiasftv.pdf`.

[3] Forrest Briggs, Raviv Raich, and Xiaoli Z. Fern. Audio classification of bird species: A statistical manifold approach. In *ICDM '09: Proceedings of the 2009 Ninth IEEE International Conference on Data Mining*, pages 51–60, Washington, DC, USA, 2009. IEEE Computer Society.

[4] Mike Brown, Dave Chaston, Adam Cooney, Dharanidhar Maddali, and Thomas Price. Recognising birds songs - comparative study. unpublished, 2009.

[5] Seppo Fagerlund. Automatic recognition of bird species by their sounds. Master's thesis, Helsinki University of Technology, 2004.

[6] Seppo Fagerlund. Bird species recognition using support vector machines. *EURASIP J. Appl. Signal Process.*, 2007(1):64–64, 2007.

[7] T. Q. Gentner. Temporal scales of auditory objects underlying birdsong vocal recognition. *Acoustical Society of America Journal*, 124:1350–1359, 2008.

[8] Richard L. Hutto and Ryan J. Stutzman. Humans versus autonomous recording units: a comparison of point-count results. *Journal of Field Ornithology*, 80(4):387–398, December 2009.

[9] Joseph A. Kogan and Daniel Margoliash. Automated recognition of bird song elements from continuous recordings using dynamic time warping and hidden markov models: A comparative study. *The Journal of the Acoustical Society of America*, 103(4):2185–2196, 1998.

[10] Chang-Hsing Lee, Yeuan-Kuen Lee, and Ren-Zhuang Huang. Automatic recognition of bird songs using cepstral coefficients. *Journal of Information Technology and Applications*, 1(1):17–23, 2006.

[11] Panu Somervuo, A. Harma, and S. Fagerlund. Parametric representations of bird sounds for automatic species recognition. *IEEE Transactions on Audio, Speech & Language Processing*, 14(6):2252–2263, 2006.

[12] Hemant Tyagi, Rajesh M. Hegde, Hema A. Murthy, and Anil Prabhakar. Automatic identification of bird calls using spectral ensemble average voice prints. In *Proceedings of the 13th European Signal Processing Conference (EUSIPCO)*, Antalya, September 2005.

A Summary of reviewed research papers

Author(s)	Data type	Features	Algorithm	Results
Kogan & Margoliash	Calls and syllables	FFT, MFCC + 1st, 2nd derivative	DTW, HMM	HMMs generally outperformed DTW, but DTW could be made to perform slightly better with more templates.
Somervuo et al.	Syllables and songs	Sinusoidal models, MFCCs, descriptive parameters	DTW, GMM, HMM	On syllables, DTW worked as well as fixed-dimensional reprecentations. On songs, HMMs performed as well as GMMs
Lee et al.	Syllables	LPCCs and MFCCs, averaged and codebook	HMM, euclidean distance nearest neighbour	codebook-MFCCs + LDA performed best
Tyagi et al.	Calls	SEAVs, MFCC, PLPCC, Rasta-PLPCC	Nearest neighbour using euclidean distances, GMMs, DTW	GMMs using MFCCs performed best
Fagerlund	Syllables	Descriptive parameters, MFCC + 1st, 2nd derivative	SVM, nearest neighbour	SVM with mixture of features worked slightly better than other methods.
Briggs et al.	Songs	Mean frequence + bandwidth, MFCCs, spectrum density, as averages, histograms or codebooks	SVM, nearest neighbour	MFCCs as codebook histograms, with nearest neighbour and Kullback-Leibler performed best, SVM approach performed similarly.
Brown et al.	Songs and syllables	DCT cepstral coefficients	DTW, GMM, SVM and HMMs (via SongScope)	GMMs performed better than the alternatives.

Table summarizing scientific literature on Bird vocalization recognition.