Automated recording system to monitor grasshopper abundance in natural environments

ESPLAB - EPFL

Professor: Pierre-André Farine

Assistants : Sara Grassi

Patrick Stadelmann

In collaboration with: Prof. Loïc Pellissier, ETHZ

William Ducret | ESPLAB | 8 janvier 2015

Summary

1. Introduction

Orthoptera are good bio-indicators and monitoring them give us an estimation of the quality or the changes through the years of the landscape. Orthoptera are used because their songs are loud and quite different between the species, thus by looking at which species we find in a land tell us the biodiversity of this place; more polluted fields or fields near to urbanization usually have less variety of species.

The manual sampling is a long and laborious task; many biologists have to capture orthoptera, recognize them, count them, and this for different fields and for different periods of the year. So it would be more efficient to have an automatic recording system capable of recognizing the orthoptera by sound.

The final goal in the long-term is a smartphone application able to recognize species of orthoptera, so that everyone can contribute to the monitoring of the landscape.

To contribute to this development, this project consists of making a first algorithm on matlab and to study the performance of recognizing orthoptera from sounds. Using a database of orthoptera recordings (see section 3.1.2) a classification model was created, then different testing methodologies were applied to check the accuracy of the model.

As a guideline, we tried to reproduce the method described in [1]. Two methods to extract the features were used: variable frame-length and fixed frame-length. In the first case, an activity detector was implemented to extract features only when the orthoptera sings, and for the second method, a frame of fixed-length is used. The features selected were the time duration of the frame, the main frequency and the first 23 Linear Frequency Cepstral Coefficients (LFCC). Finally a combination between Gaussian Mixture Model (GMM) [2] and Probabilistic Neural Network (PNN) [2] was used to classify. For this project, except that we only used a GMM classifier instead of a combination of PNN and GMM, the same procedure was applied to see if we can obtain the same results as in [1], which is around 95% of accuracy.

This report is organized as follows: section 2 explains the features used and the classification methods. Section 3 explains the data used and the experimental work. Section 4 gives the results, and finally in sections 5 we give conclusions and propose future work.

2. Recognition system

Figure 1 gives the general block diagram of the recognition system, explained as follows:

(a) The training dataset comes from the database. Each species must have sufficient different recordings to train correctly the models.
(b) Features are extracted from the audio files to have features vectors.
(c) A GMM is created for each species, using the features vectors.
(d) A record is presented to the model to be identified. It comes from the database or from the field recordings.
(e) The features are extracted to have a features vector.
(f) The features vector is presented to each GMM model, which return the probability that the record belongs to it.
(g) Based on the probability returned by the classifier, a decision is taken to label the record. To decide that, the record belongs to the class that has higher probability.

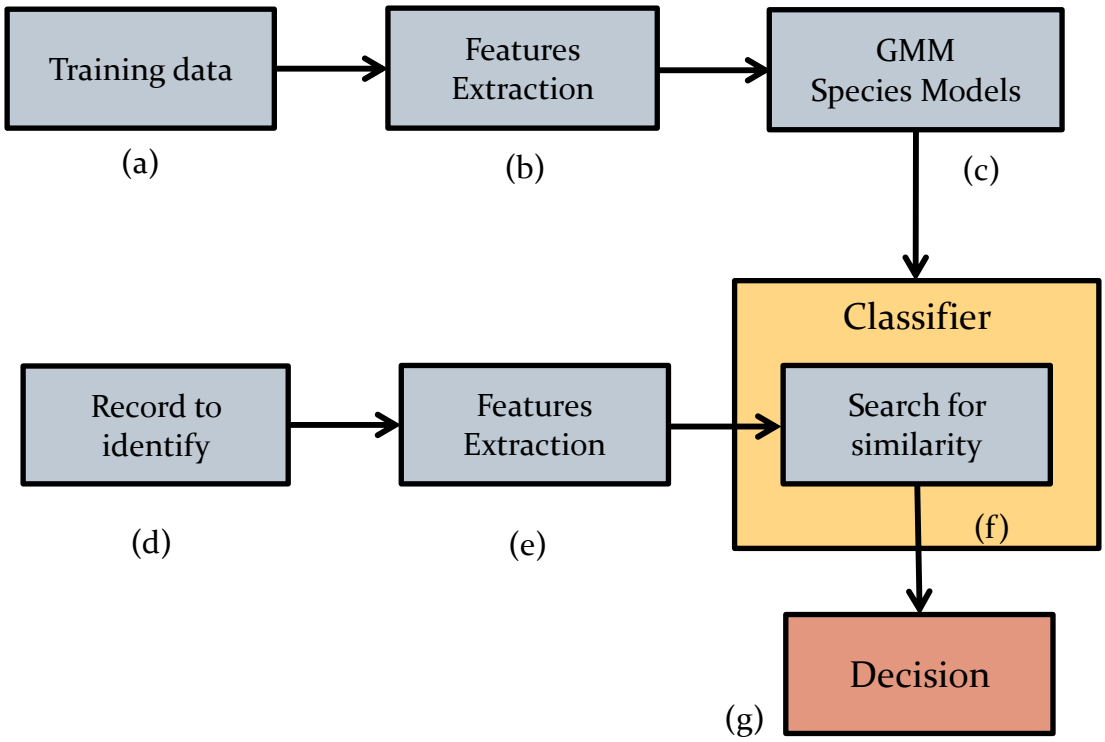

Figure 1: General block diagram of the recognition system

2.1. Features extraction

The features are the characteristics which describes the signals and allow the automatic distinction of species. A simple feature could be the main frequency (see section 3.1.3), it varies between the species, so depending on the main frequency of the record to identify, and one can determine which species it is. But as many species have a similar main frequency or a main frequency with a large variance, we should add other features that allow distinction between species.

To extract these features, one have to cut the signal into frames, then the features will be calculate for each frame. We tried two methods to cut these frames: fixed frame-length and variable frame-length, according to [1].

2.1.1. Pre-processing

Before extracting the features, few operations are applied to the signal as shown in figure 2.

Segments of all-zero values at the beginning and at the end are removed as well as segments longer than 10 ms placed within the signal. We're doing this because some files contain this unnatural segments as if they were modified after recording.

The files in the database were recorded at different sampling frequencies, all higher than 44.1 kHz. As a first step, we resample them to have all the same sampling frequency of 44.1 kHz. This means that they have a bandwidth (available frequencies) of 22 kHz, according to Nyquist-Shannon theorem, which is enough to cover the spectra of the orthoptera songs.

Some files were recorded in stereo and other in mono. In all cases we selected the first channel. Then, for each recording, we removed the mean (DC offset) and divided by the maximum of the absolute value so that the signal is in the [-1, 1] range.

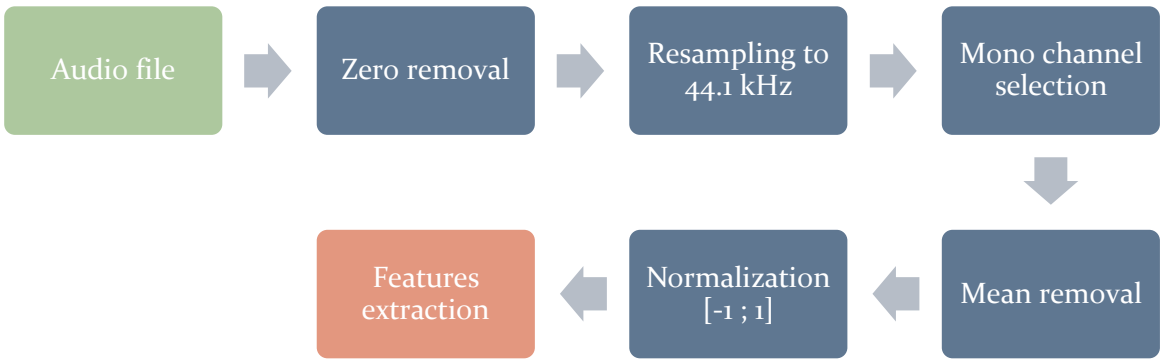

Figure 2: Overview of the pre-processing operations

2.1.2. Fixed frame-length method

In this method, we cut the signal into fixed frames of 100 ms, thus having a length of L = 4'410 samples. The frames are overlapping by 95 %, which means the frames are moving by steps of 5 ms. Then, we windowed each frames using a Hamming window.

$$W(n) = 0.54 - 0.46 * \cos\left(\frac{2\pi n}{N}\right), n = 0, \dots, L-1 \quad (1)$$

The signal is zero padded to a length of N = 8'192 samples. The STFFT (Short Time Fast Fourier Transform) is performed using:

$$X(k) = \sum_{i=0}^{N-1} x(i) * \exp\left(-\frac{j2\pi ik}{N}\right), k = 0, \dots, N-1 \quad (2)$$

Where $x$the pre-processed, windowed and zero-padded is signal and X is the resulting STFFT signal.

- Linear Frequency Cepstral Coefficient

Then the Linear Frequency Cepstral Coefficient (LFCC) are extracted. The LFCC represent the distribution of the power spectrum of a sound. To calculate them we use a filter bank with M = 218 equally spaced triangular filter, covering the frequency range of [0.1 22] kHz, as shown in figure 3, given by:

$$H_i(k) = \begin{cases} \frac{k - f_{bi-1}}{f_{bi} - f_{bi-1}}, for\ f_{bi-1} \le k \le f_{bi} \\ \frac{f_{bi+1} - k}{f_{bi+1} - f_{bi}}, for\ f_{bi} \le k \le f_{bi+1} \\ 0, for\ k > f_{bi+1}\ or\ k < f_{bi-1} \end{cases} \quad (3)$$

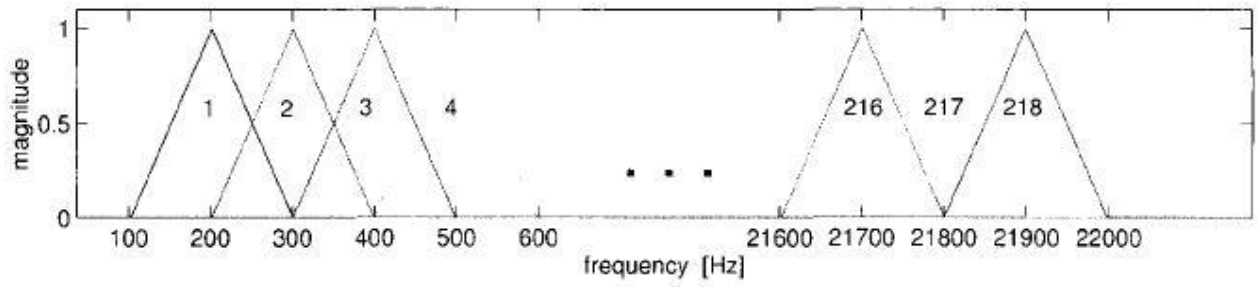

Figure 3: filter-bank spread illustration, taken from [1]

Using this triangular filters, we group the FFT coefficients:

$$\boldsymbol{X_i = \log_{10}\left(\sum_{k=0}^{N-1} |X(k)|^2 * H_i(k)\right), i = 0, \dots, M-1} \quad (4)$$

Finally we apply DCT (Discrete Cosine Transform) to de-correlate the output of the filter banks, obtaining the LFCC:

$$\boldsymbol{LFCC_j = \sum_{i-1}^{B} X_i * cos\left(j * \left(i - \frac{1}{2}\right) * \frac{\pi}{B}\right), j = 0, \dots, J} \quad (5)$$

As in [1], we retain the LFCC1 to LFCC23 removing the index 0 (energy) to avoid any dependency on the field recording setup.

Cepstral mean subtraction (CMS) is applied on a per-file basis, by subtracting the mean of the feature vector (of the file) to all LFCC:

$$\boldsymbol{LFCC_{cms} = LFCC - mean(LFCC)} \quad (6)$$

- Main frequency

The main frequency in a frame is the maximal frequency, calculated from the squared magnitude of the FFT, using the formula:

$$\boldsymbol{F0 = \frac{F_s}{N} argmax\{|X(k)|^2\}} \quad (7)$$

Where F0 is the main frequency, Fs is the sampling frequency, N is the length of the frame and X is the FFT of the signal.

Finally we assemble the feature vector by concatenating the $LFCC_{cms}$ and the main frequency, obtaining for each frame a vector of length 24. As shown in figure 4.

Frame 1: Feature vector$^{(1)}$	$F0^{(1)}$	$LFCC_{cms}1^{(1)}$	$LFCC_{cms}\dots^{(1)}$	$LFCC_{cms}23^{(1)}$
Frame 2: Feature vector$^{(2)}$	$F0^{(2)}$	$LFCC_{cms}1^{(2)}$	$LFCC_{cms}\dots^{(2)}$	$LFCC_{cms}23^{(2)}$
Frame…: Feature vector$^{(\dots)}$	$F0^{(\dots)}$	$LFCC_{cms}1^{(\dots)}$	$LFCC_{cms}\dots^{(\dots)}$	$LFCC_{cms}23^{(\dots)}$
Frame T: Feature vector$^{(T)}$	$F0^{(T)}$	$LFCC_{cms}1^{(T)}$	$LFCC_{cms}\dots^{(T)}$	$LFCC_{cms}23^{(T)}$

Figure 4: Feature vector for the fixed frame-length method

These vectors are standardized over the entire set:

$$\widetilde{\boldsymbol{Vec}_{\boldsymbol{i}}} = \frac{\boldsymbol{Vec} - \boldsymbol{\mu}_{\boldsymbol{v}}}{\boldsymbol{\sigma}_{\boldsymbol{v}}} \tag{8}$$

Where the mean μ and the standard deviation σ are calculate over the vectors of the entire dataset.

2.1.3. Variable frame-length method

In this method, the features vector is extracted very similarly as in the fixed frame-length method. Except that the frame length is not fixed but is calculated as the intervals in which there is sound activity.

In the case of a variable frame-length method, we have to implement an activity detector which tell us when the frame begins (activity) and when it ends (no activity). The decision is based on the energy and is calculated for every file as this:

a) The energy is smoothed using the matlab command `E = smooth(x.^2,Fs/10);` , where x is the preprocessed signal and Fs the sampling frequency.
b) A noise threshold is calculated using `th_noise = min(E) + .02*mean(E);`
c) Then, only the signal which is lower than this threshold is kept to calculate the energy of the noise :

```
sil = E < th_noise;
E_noise = mean(E(sil));
```

d) If `sil` is empty, we assign to `E_noise` a small default value (1e-5).
e) Then we threshold the signal to obtain the decision: activity = E > E_noise.

An example of this process is in figure 5, we see the frames represented by a pink line.

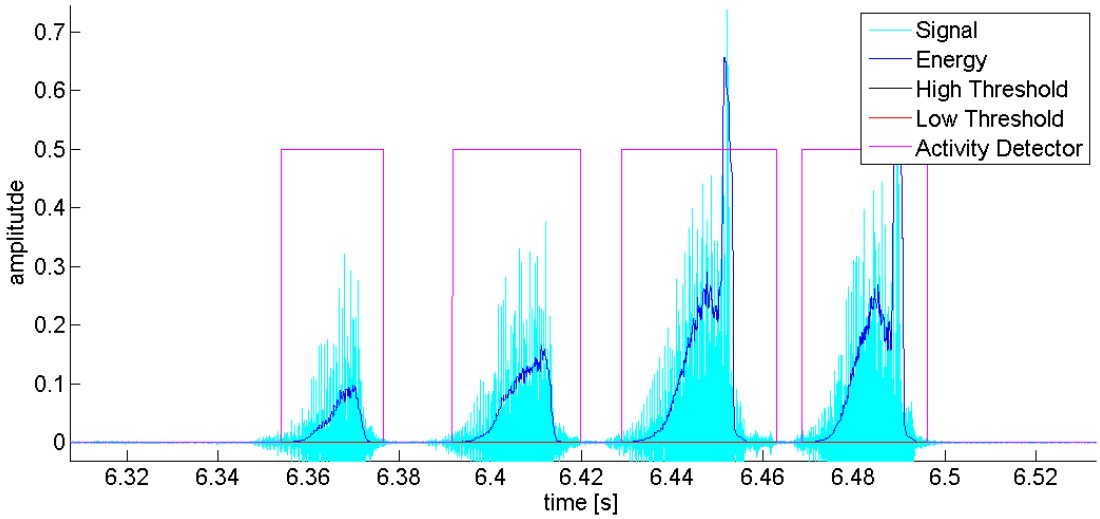

Figure 5: example of the activity detector, the frames are represented by the pink line

After the frames are cut, we calculate the mean subtracted LFCC and the main frequency (F0) as in the fixed frame-length method (see section 2.1.2) and assemble the feature vector by concatenating the LFCC, F0 and the time duration, Td, of each frame as shown in figure 6. The length of each feature vector is 25. These vectors are also standardized as explain in section 2.1.2.

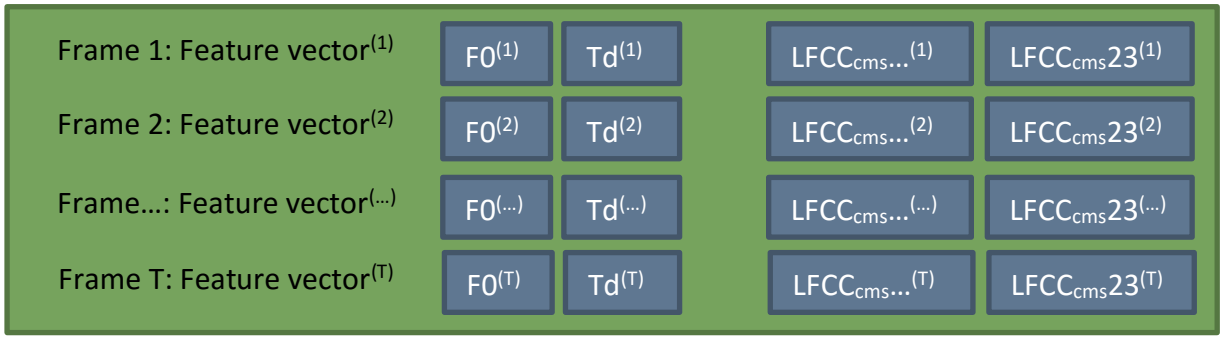

Figure 6: Feature vector for the fixed variable-length method

2.2. Classification

Now that we have all the feature vectors extracted from the files, we can perform the classification. We tried naïve Bayes classification as a first basic classifier. Then, we implemented a GMM (Gaussian Mixture Model) classifier to improve the results and have a more general classifier.

2.2.1. Naïve Bayes

Naïve Bayes is a simple classifier which (naively) assumes that the predictors are conditionally independent, given the class. Naive Bayes classifiers assign observations to the most probable class (in other words, the maximum a posteriori decision rule). This classifier was implemented using the matlab command "fitcnb" [3].

The Bayes formula is:

$$\boldsymbol{P(\omega_i|\vec{x}) = \frac{P(\vec{x}|\omega_i)P(\omega_i)}{P(\vec{x})}} \qquad (9)$$

Where $\vec{x}$ is the feature vector. of dimension Dx1, calculated in section 2.1.1 and 2.1.2, and ω_i is the class (species).

$P(\vec{x}|\omega_i)$ is the probability density of the feature vector $\vec{x}$ for each class ω_i, $P(\omega_i|\vec{x})$ is the a-posteriori probability (the probability that $\vec{x}$ belongs to ω_i) and $P(\omega_i)$ is the a-priori probability for each class (usually classes have the same probability).

$P(\vec{x})$ is the total probability:

$$\boldsymbol{P(\vec{x}) = \sum_{i=1}^{N_c} P(\vec{x}|\omega_i)P(\omega_i)\,, \qquad N_c = number\ of\ classes} \tag{10}$$

The probability density $P(\vec{x}|\omega_i)$ is approximated by a multivariate Gaussian distribution:

$$\boldsymbol{P(\vec{x}|\omega_i) = \frac{\exp[-\frac{1}{2}(\vec{x} - \overrightarrow{\mu_i})^t \mathbf{C}_i^{-1}(\vec{x} - \overrightarrow{\mu_i})\,]}{(2\pi)^{D/2}\sqrt{\det(C_i)}}} \tag{11}$$

D is the dimension of the feature vector, $\overrightarrow{\mu_i}$ is the mean vector (of dimension Dx1) and C_i is the covariance matrix (of dimension DxD) defined as follow:

$$\boldsymbol{\overrightarrow{\mu_i} = \frac{1}{K_i}\sum_{k=1}^{K_i} \vec{x}_k} \tag{12}$$

$$\boldsymbol{C_i = \frac{1}{K_i - 1}\sum_{k=1}^{K_i} (\vec{x}_k - \overrightarrow{\mu_i})(\vec{x}_k - \overrightarrow{\mu_i})^t} \tag{13}$$

The terms on the diagonal of C_i are the variances in each dimension and the other terms are the cross-variances.

Note that in the case of the Naïve Bayes assumption, in which the features (predictors) are independent, the cross-variance terms are equal to zero, so the covariance matrix is diagonal.

Once we have calculated the probability density $P(\vec{x}|\omega_i)$ for each class using (11), we can calculate the a-posteriori probability using (9). That is the probability of belonging to a class (species) having observed a feature vector.

After calculating the a-posterior probability for each class, we label the feature vector as belonging to the class yielding the maximum a-posterior probability.

2.2.2. Gaussian Mixture Model (GMM)

In GMM, we model the probability density function of the feature vector $\vec{x}$ for each class ω_i as a sum of weighted multivariate Gaussians:

$$\boldsymbol{p(\vec{x}|\omega_i) = \sum_{m=1}^{M} \alpha_m g_m(\vec{x})} \quad (14)$$

Where each g_m is a multi-variate Gaussian calculated as in (11), (12) and (13) and α_m is the weight of each Gaussian.

The training of the GMM models is done using the expectation maximization algorithm [2] [4]. As in section 2.2.1, we label the feature vector as belonging to the class yielding the maximum a-posterior probability.

The GMM classifier was implemented using the matlab command "fitgmdist" [5]. One model per species was created. Using the arguments of "fitgmdist", one can specify the type of the covariance matrix (diagonal or full) and the number of Gaussians (M).

3. Experimental work

3.1. Data

The data are recorded song files of orthoptera species. Each files contains one individual of a specific species. To train the classifier, we need to have sufficient files of each species to correctly train the models.

We will separate the data into two sets: the training set and the testing set. The training set will contain several recordings for the same species, it will be used to train the classifier. The testing set will be used to test the model and calculate its accuracy. It will contain either files from the database or recordings taken from the field.

3.1.1. Field recordings

Orthoptera is an order from the insect class, it has two main sub-orders: the Ensifera and the Caelifera. The Ensifera covers the crickets and the katydids (bush crickets) while the Caelifera covers the grasshoppers.

On the 3th of august 2015, field recordings were taken at Aigle (VD, Switzerland). They could be used as testing set for the final test (see section 6) to see if we can distinguish recordings that come directly from the field.

After, these recordings were heard and visualized using CoolEdit [6] to produce files that contain meaningful sounds. Then we gave them to an entomologist who classified them by species. There were seven species, given in figure 7, and we selected them for this study. Their names are indicated by their gender and species name, or sub-gender and species name if a sub-gender exist.

Orthoptera	Ensifera	Platycleis Albopunctata Albopunctata
		Roeseliana (*Metrioptera*) Roeselii
	Caelifera	Stauroderus Scalaris
		Chorthippus Biguttulus
		Chorthippus Parallelus
		Gomphocerippus Rufus
		Stenobothrus Lineatus

Figure 7: classification of the seven selected species

3.1.2. Database

The main database of records come from the book “Die Stimmen der Heuschrecken” [7]. This book has a CD which contains 965 recordings for 89 species (the files into “Allgemeiner Teil_wav”, “Bestimmungsteil_wav” and “Oszillogramme_wav” are grouped into one unique folder). For each species, there is many type of songs:

- S : Calling song
- W : Courtship song
- R : Rivalry song
- A : "Alternating" or "synchronizing" the song
- US : Recorded with an ultra-sound device

Only the “S” files (standard calling song) were used to train and test the models for this project.

3.1.3. Analysis of the signal

The songs and the signals among the orthoptera are characteristic of each species and an entomologist is capable of recognizing a species just by hearing its song.

Here are three examples of signals produced by three different species. One can see that the time signal (see figure 8) is pretty different between the species: the Metrioptera Roeselii is emitting continuously contrary to the Chortippus Parallelus which is emitting pulses. By looking at the spectrogram (see figure 9), one can that see that the frequencies between the species are varying too: the Metrioptera Roeselii is emitting in the high frequencies (15 kHz - 18 kHz) contrarily to the Stauroderus Scalaris which is emitting in lower frequencies (5 kHz – 10 kHz).

After studying all the seven selected species, it seems that the Ensifera are emitting in higher frequencies than the Caelifera, so it could be a first criteria to determine if the record to identify belongs to the Ensifera or the Caelifera.

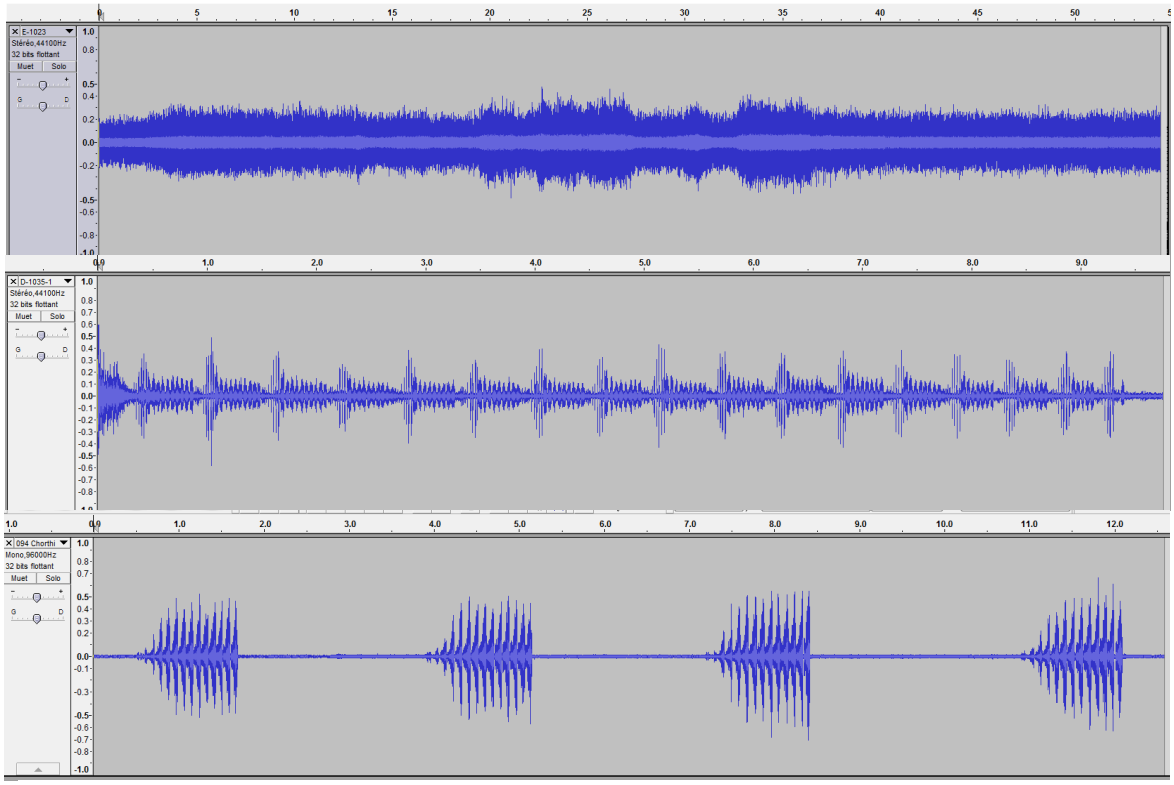

Figure 8: time domain signal, from top to bottom: Metrioptera Roeselii, Stauroderus Scalaris, Chortippus Parallelus

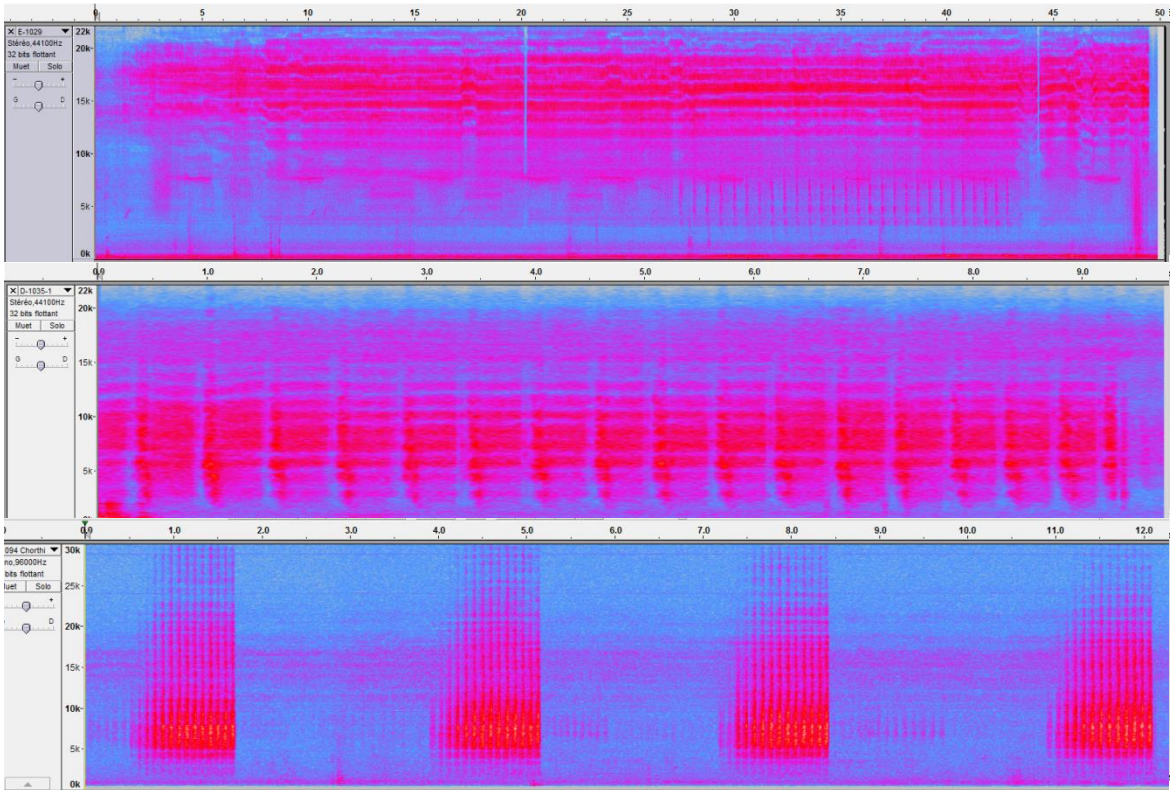

Figure 9: spectrogram, from top to bottom: Metrioptera Roeselii, Stauroderus Scalaris, Chortippus Parallelus

3.2. Test / Validation

Now that we have a model, we have to test its performance and calculate its accuracy. To do this, we perform a cross-validation or validation test to calculate the accuracy of the system. Different methods were tested, leading to different results depending on the procedure used. These methods consists of creating a training set and a testing set. The training set will be used to train the classifier models, and the testing set will be used to test the classifier models.

Each feature vector of the testing set are grouped by individual and are presented to each model. To label the individual (decide to which species it belongs), we look at the probability returned by the models and decide that it belongs to the species whose model has returned the highest probability. At the end, the accuracy is the ratio between the number of well labeled individual and the total number of individuals.

3.2.1. All-versus-all method

In the "all-versus-all" method (figure 10), all the files are used to train the models, then all the files are tested one by one. That means the same files are used to train and to test, the results are distorted. It's easier to recognize a file which has already been used to train, thus it leads to a very good accuracy which is not the representative of the reality.

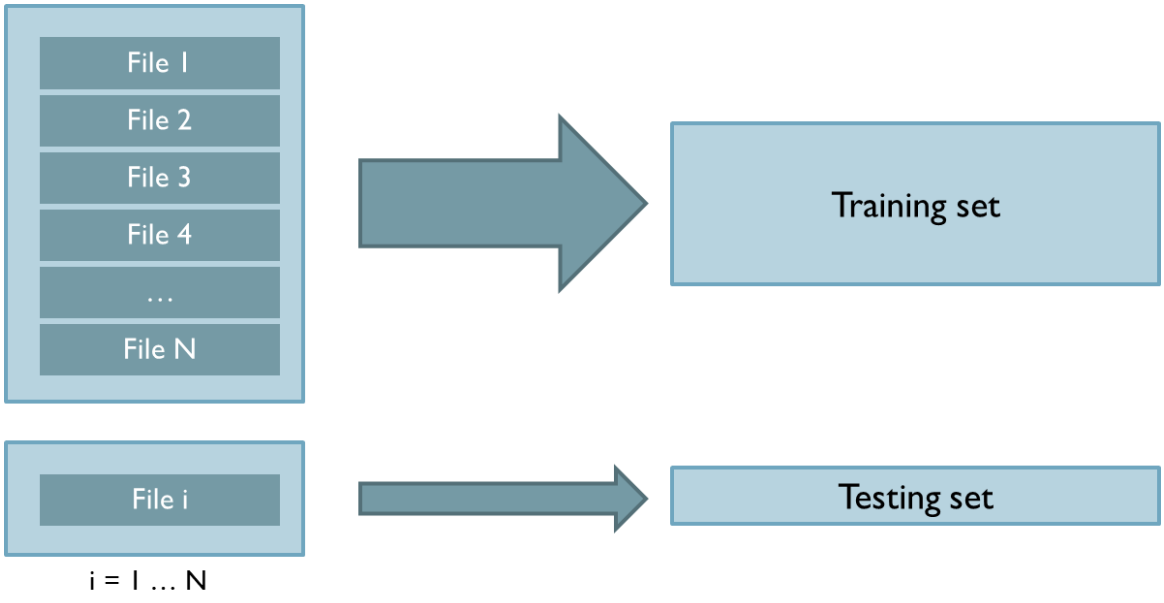

Figure 10: "All-versus-all" testing method

3.2.2. Cross-validation method

In the "cross-validation" method (figure 11), all the feature vectors are extracted from all the files and are mixed. Then, 4/5 of the feature vectors are used to train the models and 1/5 to test them. Which means that the feature vectors used to test are not used to train, but, due to the mix, a part of a file can be used to train and another part to test (a file is not used entirely to train or entirely to test).

This method is better than the previous one because we don't use the same feature vectors to train and to test. But because the training and testing sets are not totally independent (features from the same individual are used in the training and testing), it leads to a too good accuracy and is again not representative of the real condition.

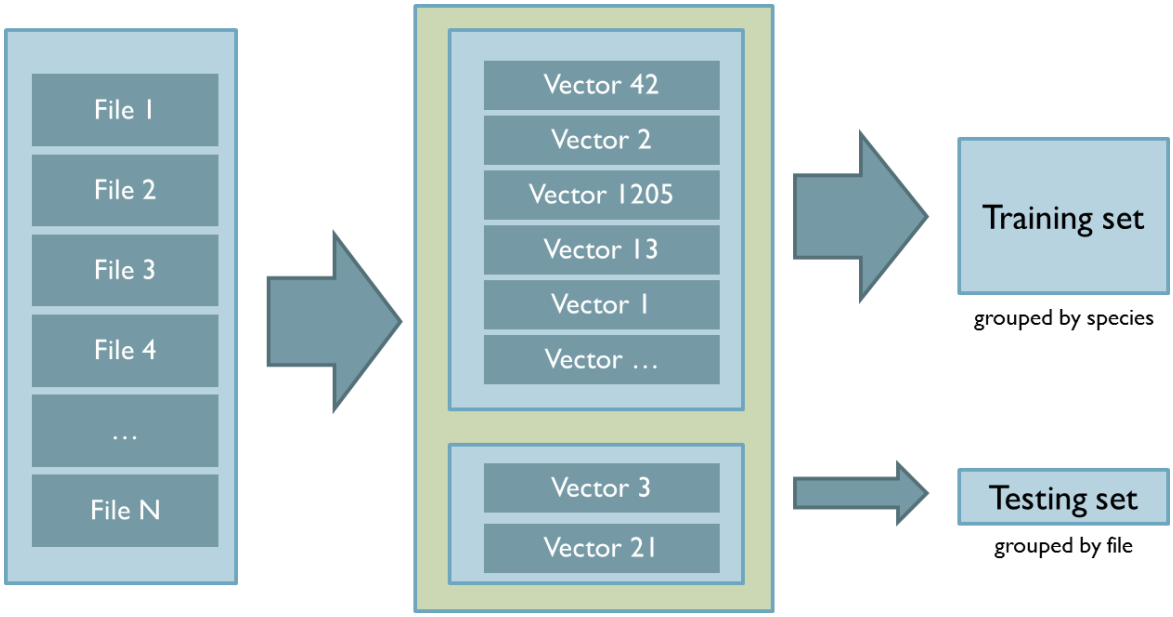

Figure 11: "Cross-validation" testing method

3.2.3. One-versus-all method

For the "one-versus-all" method (figure 12), the models are trained with all the files except for one which is removed and form the testing set. Now we have no connections between the training and testing set, the feature vectors of the file used to train are not used to test.

This operation is repeated for all the files (or individuals) to test them all. This method is close to the reality, where we train all the models and then present an unknown individual to the models. Due to this condition, the accuracy decreases drastically (see Section 4).

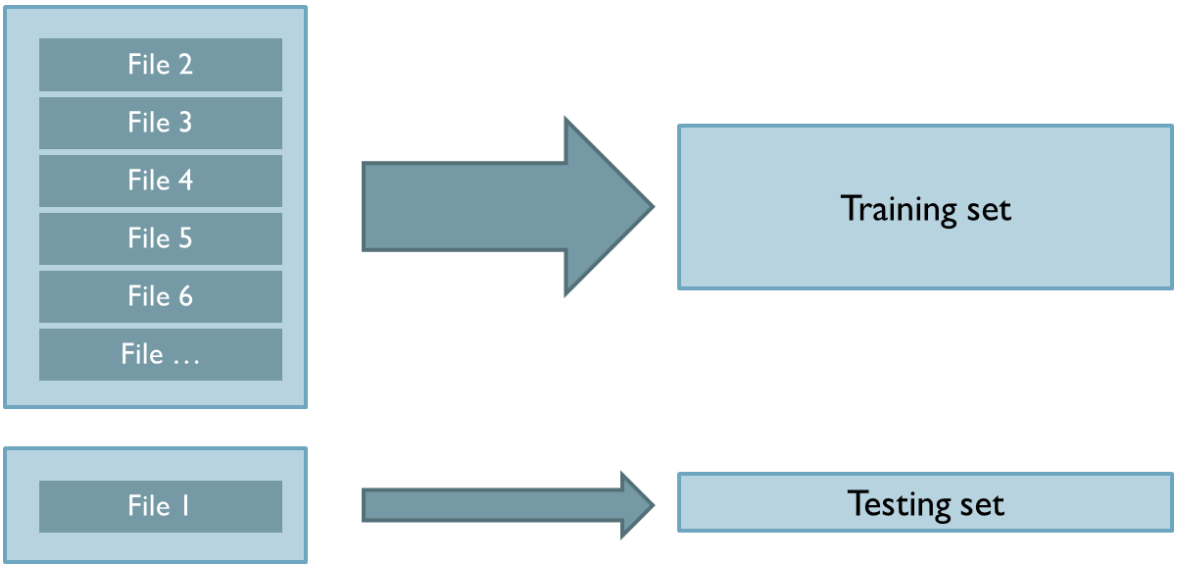

Figure 12: "One-versus-all" testing method

4. Results

One can see that the accuracy decreases drastically when we use the "one-versus-all" method (see table 2). This is due to the total independence between the training and testing set. As mentioned in section 3.2.3, the "one-versus-all" method is the one which represents more the real field condition, it is a bad hope to think that the results with the two others methods are representative.

The following results are obtained with the seven selected species (see section 3.1.1). Unfortunately, not the same amount of files per species were available, and only from three to six files per species were available. We used a diagonal covariance matrix and a number of Gaussians of K = 8. The features used are the files "data_fixed.mat" and "data_variable.mat".

Table 1: Species used for the experience leading to the results of table 2

Platycleis albopunctata albopunctata	4 files
Chorthippus biguttulus	6 files
Stenobothrus lineatus	4 files
Chorthippus parallelus	4 files
Metrioptera roeselii	4 files
Gomphocerippus rufus	3 files
Stauroderus scalaris	3 files

Table 2: Accuracy depending of the testing method used

	Fixed frame	Variable length
All versus all	100 %	100 %
Cross validation	100 %	98.6 %
One versus all	41.9 %	61.6 %

Sometimes during the execution of the matlab test file, a warning appear (Warning: Failed to converge in 100 iterations for gmdistribution with 8 components), which indicate that the model failed to converge during the training. In this case, the results can vary a little bit. The results indicated in table 2 are the mean of the accuracy of four executions.

4.1. Adding more data

Because of the lack of data (see table 1), eight other species with more data available were selected, having each six files per species. . We used a diagonal covariance matrix and a number of Gaussians of K = 8. Note that the accuracy increases when increasing the amount of files per species (see table 3 and 4). The features used are the files "data_fixed_6.mat" and "data_variable_6.mat".

Table 3: Species used for the experiment leading to the results of table 4

Chorthippus biguttulus	6 files
Tettigonia cantans	6 files
Polysarcus denticauda	6 files
Gryllotalpa gryllotalpa	6 files
Pteronemobius heydenii	6 files
Chorthippus parallelus	6 files
Psophus stridulus	6 files
Omocestus viridulus	6 files

Table 4: Accuracy depending of the testing method used with more data

	Fixed frame	Variable length
All versus all	100 %	100 %
Cross validation	100 %	100 %
One versus all	69.8 %	80.7 %

Sometimes during the execution of the matlab test file, a warning appear (Warning: Failed to converge in 100 iterations for gmdistribution with 8 components), which

indicate that the model failed to converge during the training. In this case, the results can vary a little bit. The results indicated in table 4 are the mean of the accuracy of four executions.

4.2. Confusion matrix

The confusion matrix is a representation of the well classified individuals. On the top are the true labels of the species and on the left are the labels predicted by the classifier.

For example, one can see in table 5 that the Tettigonia cantans is classified 50 % of the time correctly, 17% of the time as Polysarcus denticauda and 33% as Omocestus viridulus. So there is a little confusion with this species.

Because of the "failed to converge" matlab error, the confusion matrix can vary a little bit. The table 5 is the confusion matrix when using the fixed frame-length method (when we got 68.75 %) and the table 6 is when using the variable frame-length method (when we got 79.16 %). In both cases we used "one-versus-all" validation, a diagonal covariance matrix and a number of Gaussians of K = 8.

By looking at the two confusion matrices, it seems that the Chortippus parallelus has recognition problems. Possible reasons may be:

- The features chosen do not allow discrimination between these species (the variance within individuals of the same species is equivalent to the variance between individuals of different species).
- The amount of data is not enough for producing good models.
- There is a problem with some of the recording of the species (mismatch of recording conditions, other songs, background noise).

Table 5: Confusion matrix using the fixed-frame length method

predicted label	Chorthippus biguttulus	Tettigonia cantans	Polysarcus denticauda	Gryllotalpa gryllotalpa	Pteronemobius heydenii	Chorthippus parallelus	Psophus stridulus	Omocestus viridulus
Chorthippus biguttulus	0,33	0,00	0,00	0,00	0,00	0,00	0,17	0,50
Tettigonia cantans	0,00	0,50	0,17	0,00	0,00	0,00	0,00	0,33
Polysarcus denticauda	0,00	0,00	0,83	0,00	0,00	0,17	0,00	0,00
Gryllotalpa gryllotalpa	0,00	0,00	0,00	1,00	0,00	0,00	0,00	0,00
Pteronemobius heydenii	0,00	0,00	0,17	0,00	0,67	0,00	0,00	0,17
Chorthippus parallelus	0,00	0,00	0,00	0,00	0,00	0,33	0,50	0,17
Psophus stridulus	0,00	0,00	0,00	0,00	0,00	0,00	1,00	0,00
Omocestus viridulus	0,00	0,00	0,00	0,00	0,00	0,17	0,00	0,83

Table 6: Confusion matrix using the variable frame-length method

predicted label	Chorthippus biguttulus	Tettigonia cantans	Polysarcus denticauda	Gryllotalpa gryllotalpa	Pteronemobius heydenii	Chorthippus parallelus	Psophus stridulus	Omocestus viridulus
Chorthippus biguttulus	0,67	0,00	0,00	0,00	0,00	0,33	0,00	0,00
Tettigonia cantans	0,00	0,67	0,00	0,00	0,00	0,33	0,00	0,00
Polysarcus denticauda	0,00	0,00	1,00	0,00	0,00	0,00	0,00	0,00
Gryllotalpa gryllotalpa	0,00	0,00	0,00	1,00	0,00	0,00	0,00	0,00
Pteronemobius heydenii	0,00	0,17	0,00	0,00	0,67	0,00	0,17	0,00
Chorthippus parallelus	0,00	0,00	0,00	0,00	0,00	0,67	0,00	0,33
Psophus stridulus	0,00	0,00	0,00	0,00	0,00	0,17	0,83	0,00
Omocestus viridulus	0,00	0,00	0,00	0,00	0,17	0,00	0,00	0,83

5. Analysis and conclusion

By observing the table 2 and 4, we see that the variable frame-length method significantly increase the performance (by up to 10 %). On the other hand, the fixed frame-length method is easier to implement because we don't have to tune the activity detector, which is besides not robust to noise and varying recording conditions. The addition of further features that improve discrimination should be investigated.

Results can be distorted depending on the choice of the test methodology. Using the same data to train and test the models is non-sense, the models will recognize the data used to train it.

The testing method "one-versus-all" is the "real condition" method, which shows the generalization ability of our system. It is closer to the field conditions in which the models are already trained and an unknown record is presented to them. However, this "one-versus-all" method leads to a poor performance when few data is available (see section 4).

A larger amount of files per individuals would be needed for training good models, at least around fifteen files. If we have three files for an individual, which means two files would be used for training and one for testing, the differences between individuals within a species are not "averaged out" and removed.

That's what we see in section 4.1 before when we added more files per species, the accuracy increased by 20 %. As a future work, cleaner and reliably labelled data should be obtained and we should have more recorded files per species.

To try to improve the results, the addition of other features could be studied, e.g. using the "OpenSmile" software [8]. The Filter banks can be adjusted or wavelet analysis can be tried. Also other classifier should be investigated such as: PNN (probabilistic neural network) [2] and HMM (hidden Markov model) [2].

6. Annex: Matlab code description

- **Fixed frame-length / Variable frame-length features extraction**

To process features extraction, use the scripts “main_variable_length.m” or “main_fixed_frame.m” according to method that you want to use:

a) The database have to be in a specific architecture. First, one needs to group the clean records by species into a folder name “data_*” (see figure 13). All these folders are grouped into a main folder (“enregistrements”).
b) Open “main_variable_length.m” or “main_fixed_frame.m” and specify the path of the database (the main folder) with “base_dir” and the name of the output data files with “output_dir”.
c) Run the script, the feature vectors are saved into a “.mat” file.

- **Tests**

To train and process to perform the tests:

a) Open one of the files corresponding to the method you are interested about (“oneVsAll.m”, “allVsAll.m” or “crosVad.m”. The *_resamp files resample the frames to have the same amount among each species).
b) Change the name of the data extracted (load the .mat file with “load data_variable.mat;”)
c) Change the parameters of the GMM using the variable “cov_type” for the type of the covariance matrix and “K” for the number of Gaussians.
d) Run the script

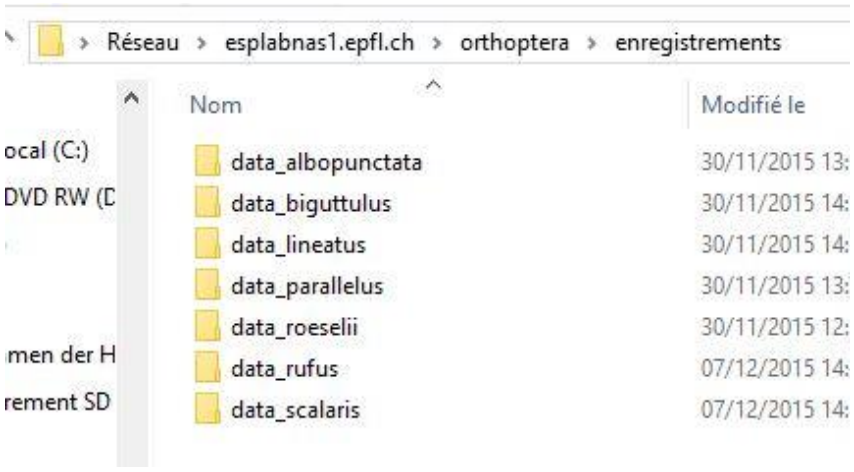

Figure 13: Structure of the database directories

To test some specific records, use the script “GMM_final_test.m” into the folder “fixed_&_variable” as follow:

a) Fill the variable “base_dir” with the path of the folder that contains the test files.
b) Check that the name of the loaded file is the data that we want to use (“load data_variable.mat;”).
c) Run the script

Note that the file “activity_detector.m” can be used to study the cutting of the frames by the activity detector. To use it, run `activity_detector(audioFile, showPlot)`, where “audioFile” is the path to the record that need to be studied and “showPlot” is a parameter to show a plot of the signal (put it to 1).

There is another version of the fixed frame-length which automatically search the files in a database according to a list of wished species. To process the fixed frame-length features extraction, use the scripts into the folder “new_fixed” as follow:

a) Specify the name of the species that need to be extracted into the files “selectedSpecies.mat” (the name must written as in the file “Datentabelle.xls”).
b) Open “calculateFeatures.m” and specify the path to the database of [7] (see section 3.1.2) with the variable “data_dir”. This variable should point to a unique directory in which one has put all the sound files (*.wav) of the database of [7]. The file selectedSpecies.mat contain the seven selected (see section 4) species and selectedSpecies_6.mat contain the eight species with more data (see section 4.1).
c) Run “calculateFeatures.m”, the resulting feature vectors are saved in the files “data_fixed.mat”

If you use this file to extract the features, when you use the file to test (e.g. “oneVsAll.m”), you have to uncomment the lines 16 to 19 and comment the lines 22 to 24. This must be done because the feature vectors are not exactly in the same structure and one needs to adapt the index.

The file “naive_bayes.m” is given as an example of implementation to use naïve Bayes matlab function [3]. It uses “cross-validation” technique.

7. Bibliography

[1] Todor Ganchev, Ilyas Potamitis, “Automatic acoustic identification of singing insects”, Bioacoustics: The International Journal of Animal Sound and its Recording, 2007, Vol. 16, pp. 281-328

[2] Pattern recognition and machine learning, Bishop Christopher M., 2009

[3] http://ch.mathworks.com/help/stats/fitcnb.html

[4] Detection and Recognition of Impulsive Sound Signals, A. Dufaux, PhD Thesis, Faculté des Sciences de l'Université de Neuchâtel, Switzerland, 2001

[5] http://ch.mathworks.com/help/stats/fitgmdist.html

[6] https://www.adobe.com/special/products/audition/syntrillium.html

[7] C. Roesti, B. Keist, “Die Stimmen der Heuschrecken”, Haupt Berne, 2009

[8] http://www.audeering.com/research/opensmile

[9] Klaus Riede, “Acoustic monitoring of orthoptera and its potential for conservation”, Journal of insect conservation, 2, p. 217-223, 1998

[10] E.D. Chesmore, E. Ohya, “Automated identification of field-recorded songs of four British grasshoppers using bioacoustics signal recognition”, Bulletin of Entomological Research, 94, p. 319-330, 2004

[11] B. & H. Baur, C. & D. Roesti, P. Thorens, “Sauterelles, grillons et criquets de Suisse”, Haupt Berne, 2006

[12] H. Bellman, G. Luquet, “Guide des sauterelles, grillons et criquets d’Europe occidentale”, Delachaux et Niestlé, 2009