IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p-ISSN: 2278-8727, Volume 27, Issue 2, Ser. 5 (Mar. – Apr. 2025), PP 11-19
www.iosrjournals.org

Bee Sound Feature Extraction Techniques

Minh Nguyen Quang, Nguyet Do Thi Minh, Binh Nguyen Thi Thanh
Faculty Of The Information Technology, Hanoi University Of Industry, Number 298, Cau Dien Street, Bac Tu Liem District, Ha Noi, Vietnam

Abstract:
Bee sounds are an essential data source for monitoring beehives as they contain a wealth of helpful information related to the health and behavior of the bee colony. Numerous studies have reported that specific buzzing sounds depend on situations such as swarming, the absence of a queen, or detecting toxins in the air. In practice, many experienced beekeepers listen to their hives to assess the status of their honeybee colonies. A growing body of research has been focused on developing systems and methods to analyze and classify bee sound data to support beekeepers in improving the effectiveness of hive monitoring. Among these studies, machine learning (ML)- based methods are considered the most powerful techniques. When using ML methods, the input cannot be raw data. Therefore, it is necessary to apply techniques to extract meaningful features from the raw data before feeding them into ML algorithms. Moreover, the performance of machine learning algorithms heavily depends on the features of the data. Consequently, after feature extraction, selecting the most relevant features for the target variable plays a crucial role in improving the performance of these algorithms. Thus, finding a suitable feature extraction technique and selecting features that are compatible with the ML algorithm is critical to developing an effective method for bee sound analysis. In this paper, the authors conduct a study to explore several feature selection algorithms for bee sounds.

Keywords: *Bee sounds; Feature extraction; MFCC; STFT;*

Date of Submission: 10-04-2025 Date of Acceptance: 20-04-2025

I. Introduction

The analysis of bee sounds has emerged as a valuable tool for monitoring the health and behavior of honeybee colonies. Acoustic signals produced within the hive carry important information related to colony activities, such as swarming, queenlessness, and environmental stressors. Traditionally, experienced beekeepers have relied on listening to hive sounds to assess colony conditions. However, with the advancement of digital technologies and artificial intelligence, automated analysis of bee sounds offers a promising approach for non-invasive and real-time hive monitoring.

To enable such automated systems, raw audio signals must be transformed into representative features that capture the most relevant acoustic characteristics. This process, known as feature extraction, plays a critical role in the performance of downstream machine learning models used for classification and decision-making. The quality and relevance of extracted features directly influence the accuracy and robustness of predictive models.

In this context, selecting effective feature extraction techniques and identifying the most informative features are key to building reliable and efficient bee sound analysis systems. This paper explores various approaches to feature extraction and selection for bee sound data, aiming to support the development of intelligent monitoring tools for beekeeping and pollinator research.

II. Mel Frequency Cepstral Coefficient (MFCC)

Mel Frequency Cepstral Coefficient (MFCC) is a feature extraction method for audio data, introduced by Davis and Mermelstein [1]. MFCC simulates the human auditory perception process by leveraging that the human ear is more sensitive to lower-frequency sounds (below 1 kHz) than to higher-frequency sounds. This method extracts features that reflect how humans perceive speech and sound, making it particularly effective in various audio analysis tasks. Figure 1 illustrates the steps of MFCC feature extraction.

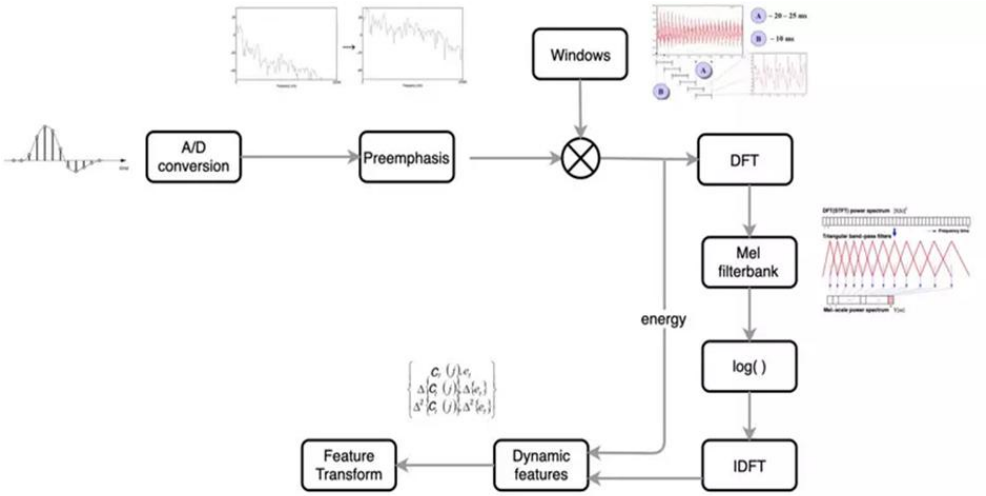

Figure 1. The steps for feature extraction of MFCC

The MFCC extraction process involves several key steps, including:

Analog-to-Digital Conversion and Pre-emphasis Filtering: Sound is a continuous (analog) signal, whereas computers operate on discrete numerical data. Therefore, it is necessary to convert the continuous signal into a discrete form by sampling it at equal intervals using a defined sampling rate, as illustrated in Figure 1. Since the human ear can perceive sounds in the frequency range of approximately 20 Hz to 20,000 Hz, the sampling rate must be high enough to preserve the vital information within this range. Typically, a sampling rate of 44,100 Hz is used, meaning 44,100 samples are taken per second. However, lower sampling rates may also be sufficient in many applications, depending on the task's requirements. Low-frequency sounds typically have high energy levels, while high-frequency sounds tend to have significantly lower energy. However, these high-frequency components often contain important phonetic information. To enhance the energy of high-frequency signals, the original signal x is passed through a pre-emphasis filter, which amplifies the high-frequency components. This process is commonly applied using the following equation:

$$y(t) = x(t) - \alpha . x(t-1) \quad (1)$$

where, $x(t)$ is the original signal at time t, $y(t)$ is the filtered output signal, α is a pre-emphasis coefficient, typically set around 0.95.

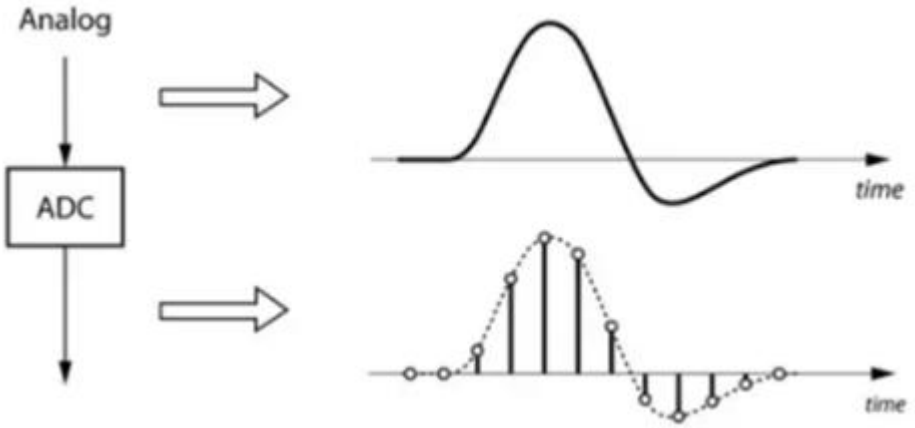

Figure 2. Illustrate analog to digital converter

Framing: This process segments the sampled audio signal into small time-based frames, with approximately 50% overlap between consecutive frames. For example, the average human speaking rate is around 3–4 words per second, with each word consisting of approximately 3–4 phonemes, and each phoneme further divided into 3–4 subcomponents. This results in roughly 36–40 segments of sound per second. Therefore, choosing a frame length of about 20–25 milliseconds is sufficient to capture one sound segment. To ensure smooth transitions and capture temporal features accurately, a typical frame overlap of 10 milliseconds is applied, as illustrated in Figure 3 below.

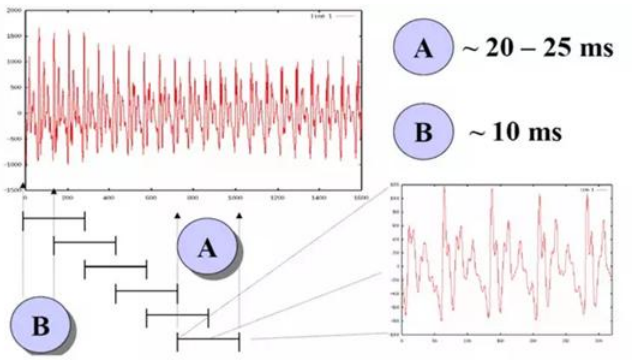

Figure 3. Illustrate the framing process [2]

Segmenting the audio signal into frames causes a sudden drop in amplitude at the boundaries of each frame. This discontinuity introduces high-frequency noise when the signal is transformed into the frequency domain. To mitigate this effect, each frame is multiplied by a Hamming window, which gradually tapers the values at the frame boundaries.

Assuming each frame contains N samples, the Hamming window is denoted by $W(n)$, where $0 \leq n \leq N-1$. The output signal $Y(n)$ from the input signal $X(n)$ after applying the window is calculated using the following formulas:

$$Y(n) = X(n) \times W(n) \tag{2}$$

$$W(n) = 0.54 - 0.46cos\left(\frac{2\pi n}{N-1}\right) \qquad with:\ 0 \leq n \leq N-1 \tag{3}$$

Fast Fourier Transform (FFT): For each frame containing N samples, the Fast Fourier Transform (FFT) is applied to convert the signal from the time domain to the frequency domain, producing the amplitude spectrum of the signal. This transformation allows for the analysis of the frequency components present in each frame, which is essential for understanding the spectral characteristics of the audio signal. The output of the FFT provides a complex-valued spectrum, but in most audio processing tasks, only the magnitude (amplitude) spectrum is used for further analysis.

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{\frac{-j2\pi nk}{N}} \qquad with: 0 \leq k \leq N-1 \tag{4}$$

Mel Filter Banks: As previously mentioned, unlike electronic measuring devices, the human auditory system perceives sound non-linearly. The human ear is more sensitive to low-frequency sounds and less sensitive to high-frequency ones. To mimic this perception, a Mel scale is constructed at this stage, and a set of triangular filter banks—typically ranging from 20 to 40 filters—is applied. These filters are used to compute the weighted sum of the frequency components so that the output approximates the human auditory response modeled by the Mel scale. The following equation is used to convert a given frequency f (in Hz) to its corresponding Mel scale value:

$$m = 2595 \log_{10}(1 + \frac{f}{700}) \tag{5}$$

Discrete Cosine Transform (DCT): This step involves transforming the log-Mel spectrum into the time domain using the Discrete Cosine Transform (DCT). The DCT compacts the signal's energy into a small number of coefficients, helping to reduce redundancy and highlighting the most relevant features. The result of this transformation is known as the Mel Frequency Cepstral Coefficients (MFCCs), which serve as a compact and informative representation of the audio signal for further analysis and classification.

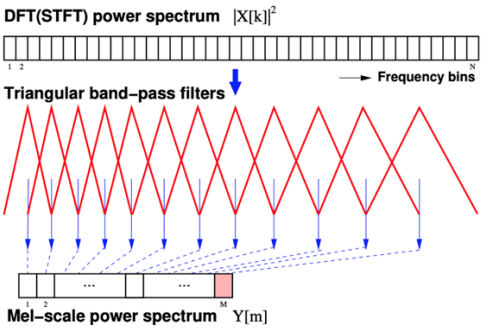

Figure 4. Example of the Mel filter [2]

III. Short-Time Fourier Transform (STFT)

The Fourier Transform is a powerful tool for signal analysis. However, it has certain limitations. When a signal is transformed from the time domain to the frequency domain, all information about time localization is lost, making it impossible to determine when specific events occur in the signal. Furthermore, the Fourier Transform is not well-suited for analyzing non-stationary signals whose frequency content changes over time.

To address these limitations, Dennis Gabor proposed an improvement known as the Short-Time Fourier Transform (STFT), which applies the Fourier Transform to short, overlapping time segments of the signal. This technique retains time and frequency information to a certain extent, making it more appropriate for analyzing non-stationary signals. As a result, it has been widely adopted in various audio and speech processing applications.

The principle of this method is to divide the signal into small segments such that the signal within each segment can be assumed to be stationary. The next step is applying the Fourier Transform to each segment. By doing so, the Short-Time Fourier Transform (STFT) preserves frequency localization due to the properties of the Fourier Transform while also providing time localization because the analysis is performed over short time windows. The following pair of equations formally define STFT:

$$\begin{cases} X_{STFT}[m,n] = \sum_{k=0}^{L-1} x[k]g[k-m]e^{-j2\pi nk/L} \\ x[k] = \sum_m \sum_n X_{STFT}[m,n]g[k-m]e^{j2\pi nk/L} \end{cases} \tag{6}$$

In this context, $x[k]$ represents the input signal, and $g[k]$ denotes a window function of length L. From the above formula, it is evident that the STFT of $x[k]$ can be interpreted as the Fourier Transform of the product $x[k]\cdot g[k-m]$, where the window is shifted along the signal. Figure 5 illustrates the computation of STFT by taking the Fourier Transforms of windowed segments of the signal. Each segment is multiplied by the window function, and then a Fourier Transform is applied to analyze the frequency content within that short time frame.

This approach applies a sliding window to the signal, and the Fourier Transform is computed on the windowed signal as the window moves along the time axis. The choice of window function is critical to the practical performance of the Short-Time Fourier Transform (STFT). Since STFT is essentially the application of the Fourier Transform to segments of the time series of interest, a key limitation is that events occurring within the window's width may not be accurately captured. In such cases, there is a lack of time resolution inherent in the Fourier Transform.

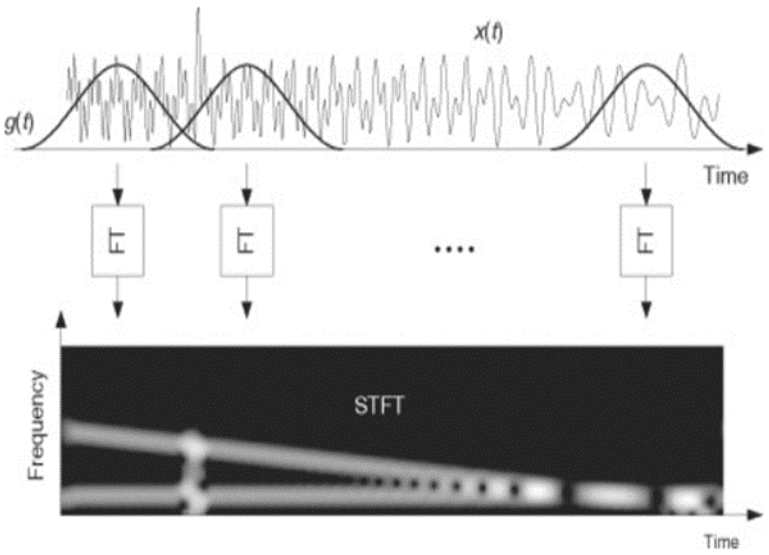

Figure 5. Illustrate the Short Fourier Transform

Due to the Heisenberg uncertainty principle, it is impossible to achieve both high time and frequency resolution simultaneously. Therefore, there exists a trade-off between time and frequency resolution in STFT. In other words, a narrow window provides better time resolution but poorer frequency resolution, and conversely, a wider window improves frequency resolution at the expense of time resolution.

The output of the STFT is often visualized using a spectrogram, a plot showing its intensity (or magnitude) over time. Figure 6 presents three spectrograms illustrating different time-frequency resolution trade-offs.

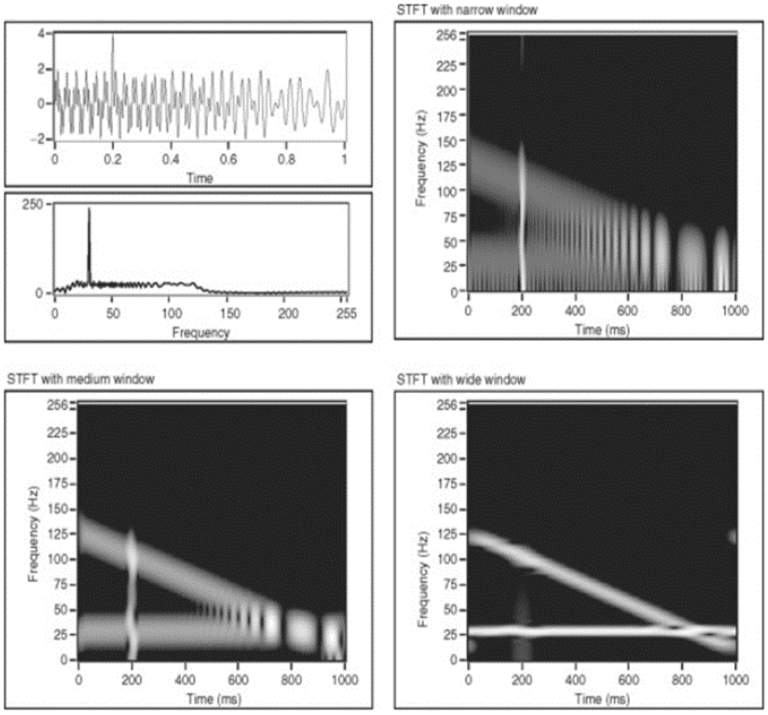

Figure 6. STFT with Different Time-Frequency Resolutions

IV. Extraction Methods

Filter method

In this method, features are eliminated based on their relationship with the output, or in other words, how they correlate with the output labels. The approach involves evaluating whether individual features have a positive or negative correlation with the target labels and subsequently removing features that are not relevant or show weak correlation with the output.

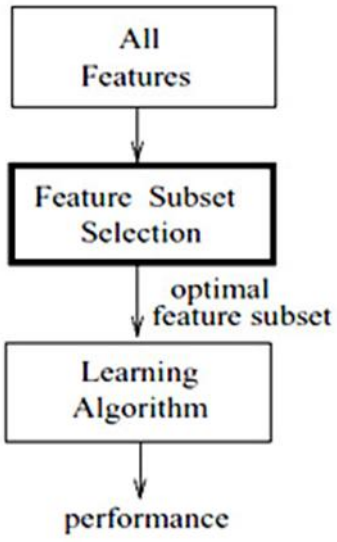

Figure 7. General flowchart of the filter method

The Filter model also consists of two main stages:
Stage 1 – Feature selection is performed using statistical measures such as information gain, distance, independence, or homogeneity, without the use of any learning algorithm at this stage.
Stage 2—This stage is similar to Stage 2 in the Wrapper model: a classifier learns from the knowledge provided by the selected features on the training dataset and is then evaluated on a test dataset.

The Filter-based feature selection model has the following characteristics:

It is independent of any specific learning algorithm (since no learning is applied in Stage 1). Still, it depends on the nature of the dataset (as it uses measures computed directly from the data). Therefore, the features selected can be reused across different machine learning algorithms.

Measures such as information gain, distance, independence, or homogeneity are typically less computationally expensive compared to evaluating classifier accuracy, making the Filter method faster in selecting relevant features.

Due to these statistical measures' simplicity and low time complexity, the Filter method is suitable for processing large-scale datasets.

However, the features selected by the Filter method do not allow the learning algorithm to adjust for errors (as they are chosen solely based on dataset-specific criteria rather than classification accuracy). As a result, the classification performance may sometimes be suboptimal.

Wrapper method

We divide the dataset into subsets and train a machine learning model. We add or remove features based on the model's output and then retrain the model. This method generates multiple feature subsets by evaluating the accuracy and performance of various possible feature combinations. Figure 8 illustrates the Wrapper model, which consists of two main stages:

Stage 1 – Feature subset selection: The best feature subsets are selected based on a classification accuracy criterion (measured on the training dataset).

Stage 2 – Learning and Testing: A classifier learns from the training data using the selected feature subset and is then evaluated on a test dataset.

As the feature subsets are generated systematically (guided by a search strategy), a separate classifier is trained on each feature subset. The classification accuracy of each model is recorded during evaluation, and the subgroup with the highest accuracy is retained. Once the feature selection process concludes, the best-performing subset is selected for final training and testing.

However, a classifier's estimated accuracy on the training set may not reflect its accurate accuracy on the test set. Therefore, determining the best performance estimate on unseen data is a key challenge. A standard solution is cross-validation, a widely adopted technique for estimating model accuracy on test datasets.

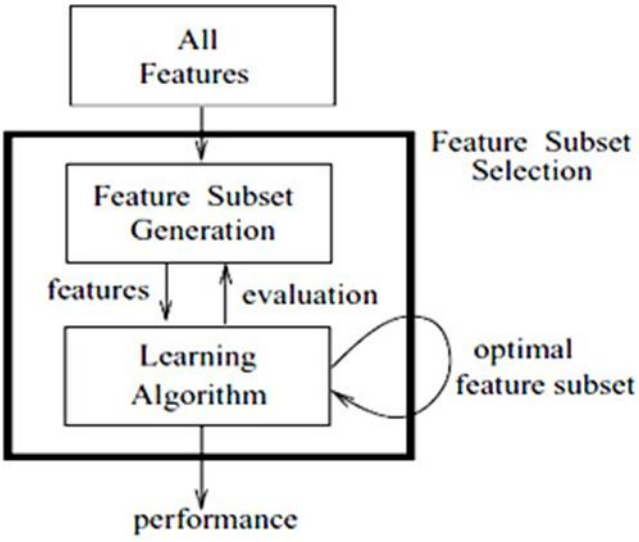

Figure 8. General flowchart of the Wrapper method

Embedded method

The Embedded method combines both of the aforementioned approaches to create an optimal set of features. Feature selection methods embedded in machine learning algorithms are typically carried out in one of two ways:

- Directly integrated into the model training process, such as with algorithms like Decision Tree or Random Forest, where feature selection is part of the model's learning process.
- Using a separate algorithm to evaluate and eliminate unimportant features, such as Recursive Feature Elimination (RFE) or SelectFromModel, which assess the features based on the model's performance.

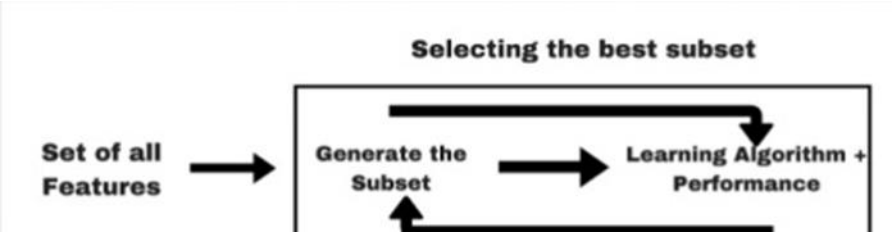

Figure 9. General flowchart of the embedded method

This method continuously trains the system, iterating repeatedly while keeping the computational cost as low as possible. In both cases, the main goal is to reduce the number of features and increase the model's accuracy. Embedded feature selection also helps to reduce overfitting and accelerate the model training process.

V. Experiment Results

We conducted experiments at the **Tropical Beekeeping Research Center**, which houses top experts in the field of beekeeping. These experts assist us with initial data labeling, specifically by identifying and removing the queen bee from the hive (simulating a **queen-less state**) and later reintroducing the queen. There are two main scenarios:

- **Normal State**: This occurs when the queen is still in the hive. We collect audio data before the queen is removed.
- **Queen-less State**: To simulate the situation of the missing queen, the research team, including the beekeeping experts, removes the queen from the hive. This scenario represents a queen-less condition.

We record the hive's sounds starting when the queen is removed and continue recording for 3-4 hours or sometimes overnight. Figure 10 illustrates the equipment used to collect sound data.

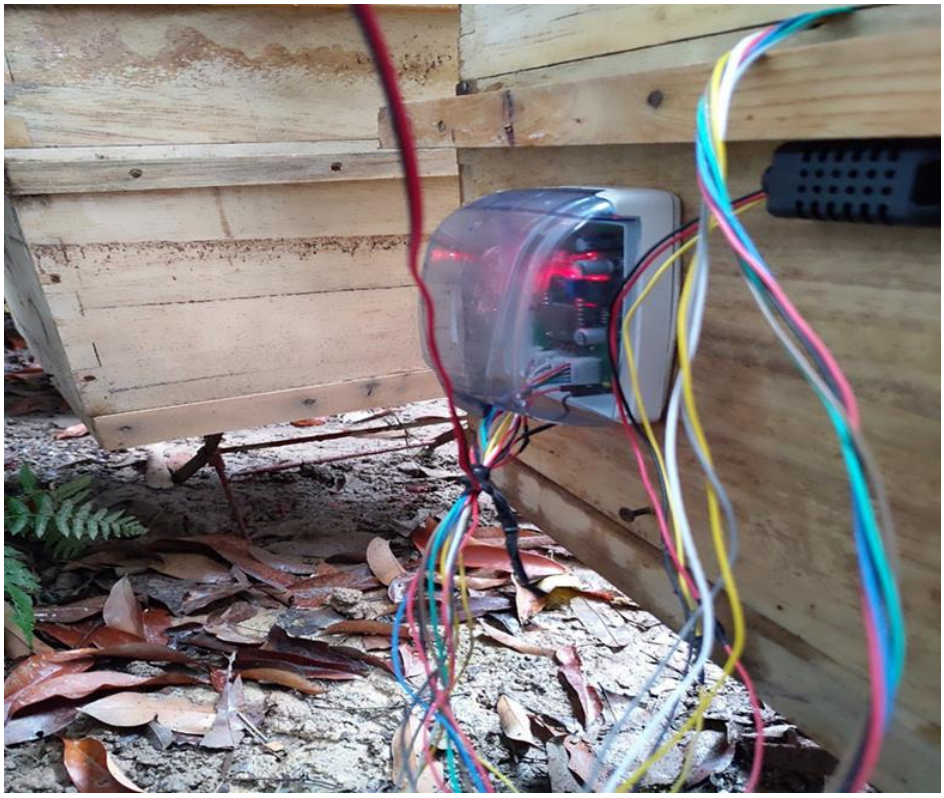

Figure 10. Equipment to collect sound data

In the study by Cecchi et al. [3], the research team collected audio data every 10 minutes. Kylunku et al. [4] collected audio data every 15 minutes for 30 seconds, then extracted 2-second segments to feed into the recognition system. In another study, Aumann et al. developed the Janus system to collect data from hives, where the research team recorded 20 seconds of audio every 2 minutes. In this study, we process the labeled audio files mentioned earlier by cutting them into shorter segments of 60s, 30s, 3s, and 2s for both the queen-less and normal states, then conduct subsequent experiments.

After performing feature extraction using the MFCC method, we apply the Embedded feature selection method, specifically the Random Forest technique. This method was developed by Breiman [5], who is also the co-author of the CART (Classification and Regression Trees) method, which is considered one of the top 10 classical data mining techniques. Random Forest is built upon three main components: (1) CART, (2) whole learning, ensemble learning, and model combination, and (3) bootstrap aggregation (bagging). Figure 11 illustrates the importance of the features provided by the Random Forest method.

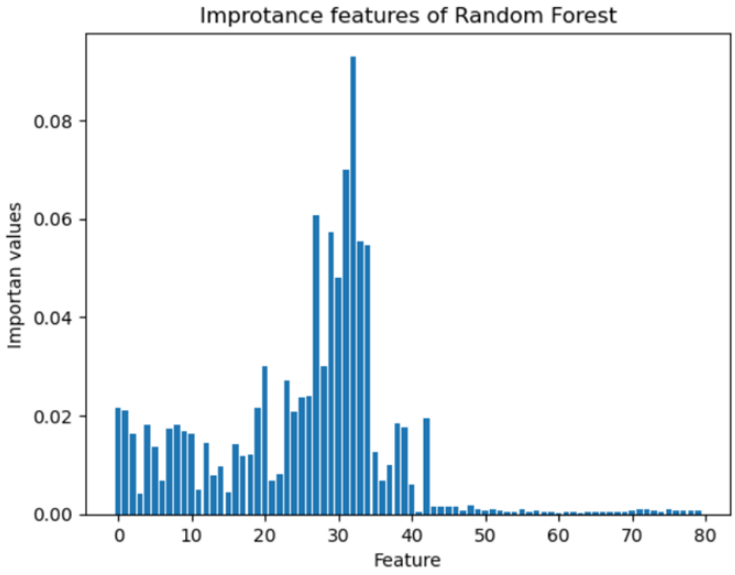

Figure 11. Feature importance generated by the RF method

We selected features with an importance greater than 0.02 based on the results. Figure 12 shows the number of features retained after applying the RF feature selection method.

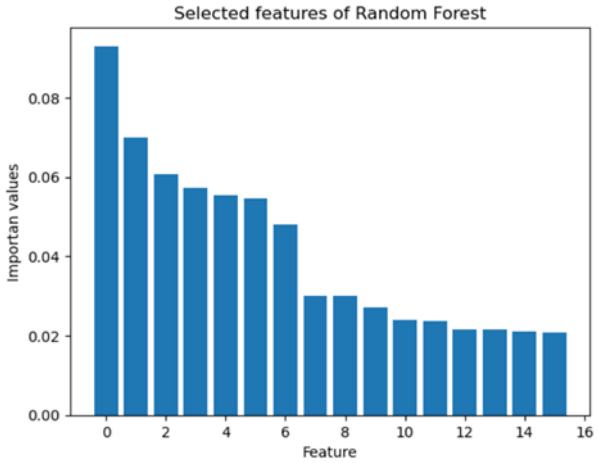

Figure 12. The number of features selected is based on the importance of the feature generated by the RF method.

VI. Conclusions

Bee sound data plays a vital role in beekeeping. Experienced beekeepers often rely on these sounds to detect abnormal situations within the hive. However, with traditional hive monitoring methods, these phenomena are usually only detected by experienced beekeepers. Therefore, modern monitoring methods using machine learning techniques and the Internet of Things (IoT) have been researched and tested to assist beekeepers in effectively monitoring their hives. To apply machine learning methods, we cannot directly use the raw audio data collected from the hive; instead, we need to extract audio features and represent them as vectors. Feature Selection is a process in machine learning where a set of the most essential features is chosen for use in machine learning algorithms. The role of feature selection is to reduce the data size, accelerate learning, and improve the accuracy of the machine learning model. It also helps prevent overfitting and reduces the model's complexity.

References

[1] Davis S. Và Mermelstein, P. Comparison Of Parametric Representations For Monosyllabic Word Recognition In Continuously Spoken Sentences, IEEE Transactions On Acoustics, Speech, And Signal Processing, 28(4): 357-366, 1980.
[2] Nguyễn Trung Thành. MFCC Cho Xử Lý Tiếng Nói. Https://Viblo.Asia/P/Feature-Extraction-Mfcc-Cho-Xu-Ly-Tieng-Noi-4dbzn2xmzym, 2020.
[3] S. Cecchi Et Al., "Multi-Sensor Platform For Real Time Measurements Of Honey Bee Hive Parameters," IOP Conf. Ser.: Earth Environ. Sci., Vol. 275, P. 012016, May 2019, Doi: 10.1088/1755-1315/275/1/012016..
[4] V. Kulyukin And S. Mukherjee, "On Video Analysis Of Omnidirectional Bee Traffic: Counting Bee Motions With Motion Detection And Image Classification," Applied Sciences, Vol. 9, No. 18, P. 3743, Sep. 2019, Doi: 10.3390/App9183743.
[5] Breiman, L., 2001. Random Forests. Machine Learning 45, 5–32.