Hindawi
Mathematical Problems in Engineering
Volume 2021, Article ID 5594498, 13 pages
https://doi.org/10.1155/2021/5594498

Research Article

Acoustic Scene Classification and Visualization of Beehive Sounds Using Machine Learning Algorithms and Grad-CAM

Jaehoon Kim,[1] Jeongkyu Oh,[2] and Tae-Young Heo [1]

[1]*Department of Information and Statistics, Chungbuk National University, Cheongju-si, Chungbuk 28644, Republic of Korea*
[2]*Data Scientist Team, BEGAS Inc, Sejong-daero 39, Jung-gu, Seoul 04513, Republic of Korea*

Correspondence should be addressed to Tae-Young Heo; theo@cbnu.ac.kr

Received 9 February 2021; Accepted 15 May 2021; Published 25 May 2021

Academic Editor: Venkatesan Rajinikanth

Copyright © 2021 Jaehoon Kim et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Honeybees play a crucial role in the agriculture industry because they pollinate approximately 75% of all flowering crops. However, every year, the number of honeybees continues to decrease. Consequently, numerous researchers in various fields have persistently attempted to solve this problem. Acoustic scene classification, using sounds recorded from beehives, is an approach that can be applied to detect changes inside beehives. This method can be used to determine intervals that threaten a beehive. Currently, studies on sound analysis, using deep learning algorithms integrated with various data preprocessing methods that extract features from sound signals, continue to be conducted. However, there is little insight into how deep learning algorithms recognize audio scenes, as demonstrated by studies on image recognition. Therefore, in this study, we used a mel spectrogram, mel-frequency cepstral coefficients (MFCCs), and a constant-Q transform to compare the performance of conventional machine learning models to that of convolutional neural network (CNN) models. We used the support vector machine, random forest, extreme gradient boosting, shallow CNN, and VGG-13 models. Using gradient-weighted class activation mapping (Grad-CAM), we conducted an analysis to determine how the best-performing CNN model recognized audio scenes. The results showed that the VGG-13 model, using MFCCs as input data, demonstrated the best accuracy (91.93%). Additionally, based on the precision, recall, and F1-score for each class, we established that sounds other than those from bees were effectively recognized. Further, we conducted an analysis to determine the MFCCs that are important for classification through the visualizations obtained by applying Grad-CAM to the VGG-13 model. We believe that our findings can be used to develop a monitoring system that can consistently detect abnormal conditions in beehives early by classifying the sounds inside beehives.

1. Introduction

Honeybees play a vital role in pollinating most food crops, which are essential for sustaining human life. According to the Food and Agriculture Organization of the United Nations, 75% of the world's top 100 crops rely on honeybees for pollination [1]. Therefore, the existence of bees is related directly to human food problems. However, the number of bee populations worldwide has declined since the mid-2000s. Experts call this phenomenon colony collapse disorder and argue that it is caused by beehive pests, abnormal climate, viruses, and fungi [2]. As reported by Chensheng et al. [3], the exposure to neonicotinoids, which is a type of pesticide, increases the mortality rate of bees. Even when exposed to nonlethal amounts of neonicotinoids, bees abandon their hives and disappear. Generally, most beekeepers install cameras inside and outside beehives to monitor and manage them directly. However, through this method, it is difficult to respond to events that occur rapidly. Moreover, this approach requires constant labor. Therefore, a novel technological approach must be developed to ensure the prompt detection of changes inside beehives.

Recently, deep learning algorithms, showing satisfactory performance in various fields, such as image classification, voice recognition, and video analysis, have been actively utilized in the field of insect study. By using deep neural networks and convolutional neural networks (CNNs) to apply short-time Fourier transform (STFT) and mel-frequency cepstral coefficients (MFCCs) extracted from sound data, Kiskin et al. [4] found mosquitoes by differentiating their

sounds from environmental sounds. By using a CNN, Rodriguez et al. [5] conducted a study on the recognition of pollinating bees in images of bees. In addition, according to the IEEE's Audio and Acoustic Signal Processing Challenge on Detection and Classification of Acoustic Scenes and Events, which annually uses audio data to solve problems of acoustic scene classification, sound event detection, and audio tagging, deep learning-based algorithms, such as CNNs, have been proposed, demonstrating exceptional performance [6–8].

To apply sound data to a deep learning algorithm, it is crucial to preprocess such data. Because audio data are high-dimensional data in which there are several frequencies and amplitudes sensitively changing with time, it is recommended to extract a feature value that reflects the signal characteristics through preprocessing. Through preprocessing, a sound signal is converted into spectral-temporal features, which can express the change in energy over time and the frequency as an image. These feature values are advantageous in that they can be easily applied to CNNs developed for image classification. In the weight computation process, such CNN models use a convolutional kernel to extract features and reduce dimensions by considering adjacent components of the feature values. In this process, energy modulation patterns based on time and frequency are effectively learned, and other sounds are identified.

Preprocessing methods are used in various manners depending on the researcher's domain knowledge and significantly affect the model's performance. Han et al. [9] focused on sound signal preprocessing and improved the overall model's performance by fusing it with a CNN model. Liu [10] extracted Gammatone cepstral coefficients, MFCCs, a constant-Q transform (CQT), and a chromagram and used them as multiple feature channels for machine learning algorithms, such as CNNs. Piczak [11] used a mel spectrogram as input data for a CNN model. Boddapati et al. [12] combined a spectrogram, MFCCs, and a cross recurrence plot and represented them as a single-color image. Su et al. [13] used multiple feature values, which combined all MFCCs, a logarithmic-mel spectrogram, chroma, spectral contrast, and the Tonnetz for analysis. The aforementioned studies were conducted based on environmental sound data, and the optimal results were obtained using different classification models depending on various preprocessing methods. Hence, it is crucial to determine the optimal classification model for each preprocessing method.

In addition, there is little insight into how the audio scene is recognized based on the result obtained by applying the preprocessed sound signal to the CNN model. However, the solution to a similar problem has proven its effectiveness in solving image classification problems. Class activation mapping (CAM) [14] was proposed as a means of emphasizing the differential image region for a specific output class of CNN models. Selvaraju et al. [15] developed gradient-weighted (Grad)-CAM, which can be applied to a wide range of CNN models. The mapping explains which parts of the image play an important role when CNN models perform classification tasks.

For beehive sound classification, only a few reported studies used CNN models by applying feature extraction methods [16, 17]. For example, Nolasco et al. [16] used the MFCCs and mel spectrograms extracted from beehive sounds through a support vector machine (SVM) and CNN models to determine whether there was a queen in the beehive; they recorded an AUC score of over 99%. In addition, Nolasco and Benetos [17] used SVM and CNN models to classify bee sounds, by extracting MFCCs and mel spectrograms from beehive sounds; they recorded an AUC score of approximately 80%. However, these studies did not explore diverse feature extraction methods, optimal parameters, and CNN models with deep layers.

In this study, we present a data preprocessing method and classification model optimized for the classification of beehive sounds by comparing the performance of various methods and models. We use a mel spectrogram, MFCCs, and a CQT as the feature extraction methods. We compare the performance of two CNN models (shallow CNN and VGG-13) to that of three machine learning methods (SVM, random forest, and extreme gradient boosting (XGBoost)). We use accuracy, precision, recall, and F1-score to measure the classification performance of the models. In addition, we present the visualization results using Grad-CAM to validate the training process of the best-performing CNN model.

This study contributes to the development of algorithms that can be applied to ensure the rapid detection and identification of changes inside beehives. This study has two main purposes. The first purpose involves using audio data to compare the performances of various machine learning algorithms and determine a suitable method for Bee/noBee classification. The second purpose involves utilizing Grad-CAM for a wide range of CNN models to explain the important factors, including the intervals of Bee or noBee, when the models predict Bee/noBee classes.

The remainder of this paper is organized as follows. Section 2 presents the motivation of this study. We present the materials and methods used for the analysis of bee classification in Section 3. In Section 4, we describe the outcomes of the comparison between some proposed methods and the visualization results of the CNN models. The discussion is presented in Section 5, and the conclusion is presented in Section 6.

2. Motivation

This study is motivated by the work of Nolasco and Benetos [17]. Nolasco and Benetos suggested that sound-based automated beehive monitoring systems improve the technological approaches to beekeeping, by identifying the phenomena related to the natural cycle of beehives. Therefore, Nolasco and Benetos explored the performance of sound-based machine learning methods. Inspired by their approach, we compare the performance of various machine learning methods in classifying the status of beehives. Moreover, through Grad-CAM, we visualize the results of the performance of various CNN models in the detection of Bee/noBee intervals.

3. Materials and Methods

This study is divided into audio identification and classification steps. The identification phase includes all the data

preprocessing procedures. The classification phase is further divided into sections using machine learning algorithms (SVM, random forest, and XGBoost) and CNN algorithms (shallow CNN and VGG-13). Figure 1 depicts a workflow that divides the entire process of this study into several steps. First, the features are extracted by dividing the audio file into small time lengths. Thereafter, in the case of the CNNs, the extracted features are used as input data to train and evaluate the models and predict the test set. The model's performance is then verified through a visualization process using Grad-CAM. In the case of the machine learning algorithms, statistics (mean and standard deviation) are obtained from the extracted features in the form of multidimensional arrays, which are used to train and evaluate the models and predict the test set. The detailed procedure of audio data preprocessing is covered in Section 3.2. The model learning and evaluation processes are covered in Section 3.

3.1. Data Annotation. In this study, we used beehive sound data from the Open Source Beehives (OSBH) project [18], which is a part of the dataset used by Nolasco and Benetos [17]. This dataset was complied using data from a general beekeeper's beehive, and the sound signals detected outside the hive were recorded. It consists of beehive sounds recorded from two beehives, which are annotated as Bee or noBee depending on the source of the sound signal, with approximately 20% annotated as noBee. The annotation procedure involved listening to the recorded sounds and marking the beginning and end of all sounds that could not be recognized as bee sounds, i.e., as noBee. All other sounds were marked as Bee. The sections annotated as noBee included cases where external sounds detected within a section were superimposed over the bee sounds. The external sounds generated at that time were mainly those of traffic, car horns, birds, and other insects. Therefore, only pure bee sounds were annotated as Bee.

The dataset consists of 19 sound files (.wav) and an annotation file (.lab) with the same name. Each file is 5 min long, with 95 min of sound files used for analysis. The sample rate of the audio files is 22,050 Hz. Table 1 shows an example of an annotation file.

3.2. Data Preprocessing. In this study, we divided the audio file into 1-second intervals and extracted three features: a mel spectrogram, MFCCs, and a CQT. When dividing the audio files, files containing both Bee and noBee intervals were treated as noBee. Because machine learning algorithms perform well only when the provided features are appropriate and relevant, the extracted features are essential for solving machine learning problems. We focused on extracting spectral features from the audio signals. Each feature was extracted using a function provided by Librosa, which is a Python library [19]. Figure 2 shows the features extracted from the audio files as images. To include more detailed information, we used the librosa.feature.delta function provided by the Librosa library to add delta1 (Δ), which is a first-order differential estimate, and delta2 (Δ^2), which is a second-order differential estimate, for each feature. The extracted features are (n, x, y)-dimensional arrays, where the x-axis represents time and the y-axis represents frequency.

3.2.1. Mel Spectrogram. The mel spectrogram is a feature value that represents sound in image form. It is mainly used in sound analysis [20]. First, the audio signal is divided into sections by frame, and the spectrum for each section is obtained through STFT. Subsequently, the audio signal in the time domain is reexpressed in the frequency domain, and the mel scale spectrum is defined as the spectrum to which the mel scale is applied. The mel spectrogram is an image of the converted mel scale spectrum. It combines a waveform, visually showing changes in the amplitude axis over time, and a spectrum, showing changes in the amplitude axis over frequency. Moreover, it represents the difference in color amplitude.

3.2.2. MFCCs. MFCCs are feature values that are widely used in audio/signal processing [21]. They are extracted by applying cepstral analysis to the mel scale spectrum, which is described in Section 3.2.1. Cepstral analysis extracts unique sound values from the spectrum. It binds the spectrum into a constant frequency band, after which it uses a logarithmic transformation and an inverse fast Fourier transform to obtain the coefficients. In this process, the correlation caused by overlapping the filter bank is separated, and a diagonal covariance matrix is created. Finally, only coefficients with a significant amount of information remain. This provides robustness against rapid signal changes and obtains the final MFCCs.

3.2.3. CQT. The CQT is a method for converting signal or sound data into frequency domain data [22]. It mainly performs well in music processing. It is similar to the Fourier transform, but it has some additional advantages. First, it uses a logarithmic scale to ensure narrow and wide bandwidths in the low- and high-frequency regions, respectively. Hence, it is more useful than the Fourier transform, which provides lower resolution in the low-frequency regions. In addition, the bandwidth is divided proportionally to the central frequency, making it easy to distinguish differences, even if the frequency spans multiple octaves.

In the case of the three machine learning algorithms (SVM, random forest, and XGBoost), the mean and standard deviation for each time section were obtained by extracting statistics from the preprocessed data and using them as input data. In the case of the CNN models, we combined each feature into a three-channel image and used it as input data. Therefore, the form of the final input data was an array of (n, x, y, 3) dimensions. The input was used to train the CNN models in the same form as the image data.

3.3. SVM. The SVM is one of the most common supervised machine learning algorithms for pattern recognition, classification, and regression [23]. Specifically, it is used for two-group classification problems. When data are in the form of

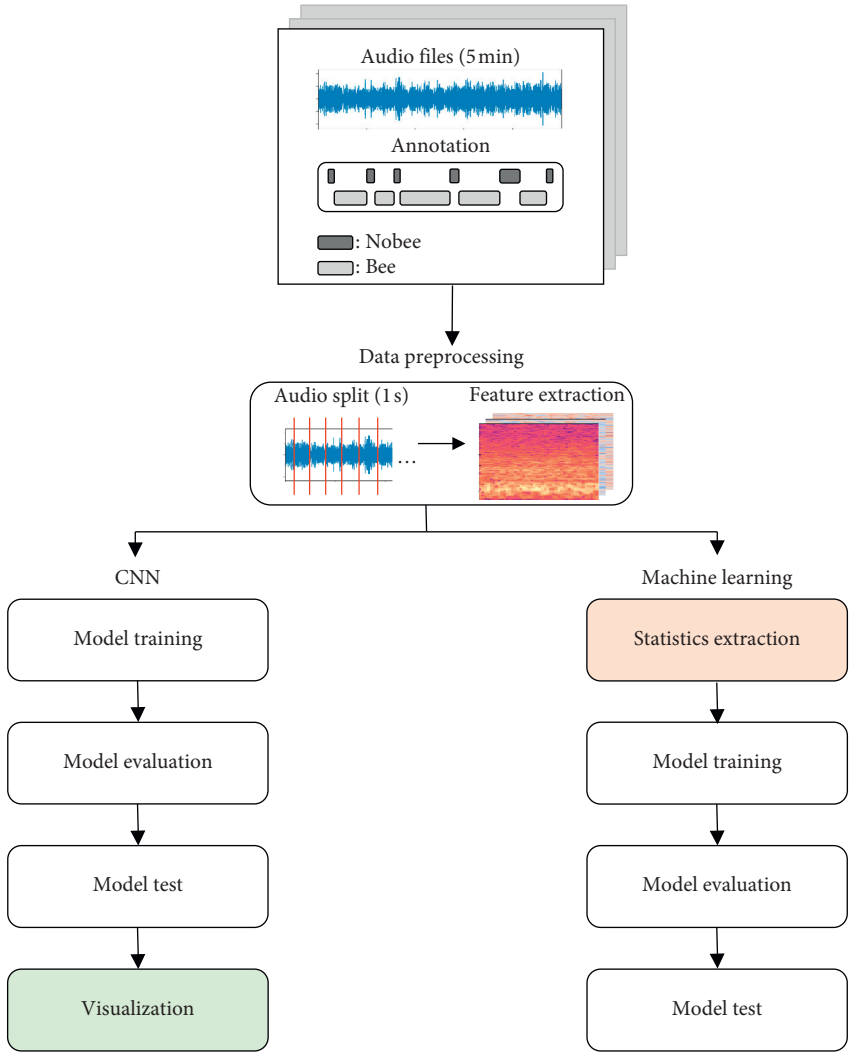

Figure 1: Workflow using two classification methodologies.

Table 1: Example of an annotation file.

Onset (s)	Offset(s)	Class label
0	7.3	Bee
7.31	7.87	noBee
7.88	10.37	Bee

a p-dimensional vector, they can be classified by dividing the dimensional space using a $p-1$-dimensional hyperplane. The main purpose of the SVM is to find the hyperplane with the largest margin among several hyperplanes.

Figure 3 shows the principle of classification using a linear SVM, assuming that data are mapped two-dimensionally. The data points of different classes closest to the hyperplane are called the support vector. The distance between the support vector and hyperplane is called the margin. When the data are not separated by a linear hyperplane, a large computational cost is required to obtain the hyperplane with the maximum margin. To solve this problem, we can use a kernel trick that replaces the dot product operation, which is required when calculating the distance between the support vector and hyperplane using a nonlinear kernel function. In this study, we use the Gaussian radiation basis function as the kernel function.

3.4. Random Forest. Random forest is an ensemble learning method used for classification and regression analysis. It is a method for collecting classification results by constructing several decision trees during the training process to obtain a conclusion [24]. In other words, each classifier is used individually to predict an independent classification result, after which it votes to predict the best result. Although the results of several decision trees may result in overfitting, because the prediction is based on numerous decision trees, the influence of overfitting is reduced, and satisfactory generalized performance is demonstrated. Figure 4 shows the general structure of a random forest.

3.5. XGBoost. XGBoost is a decision tree-based ensemble learning algorithm that uses a gradient boosting framework. Contrary to random forest, XGBoost uses a gradient descent algorithm to minimize the loss function of the previous

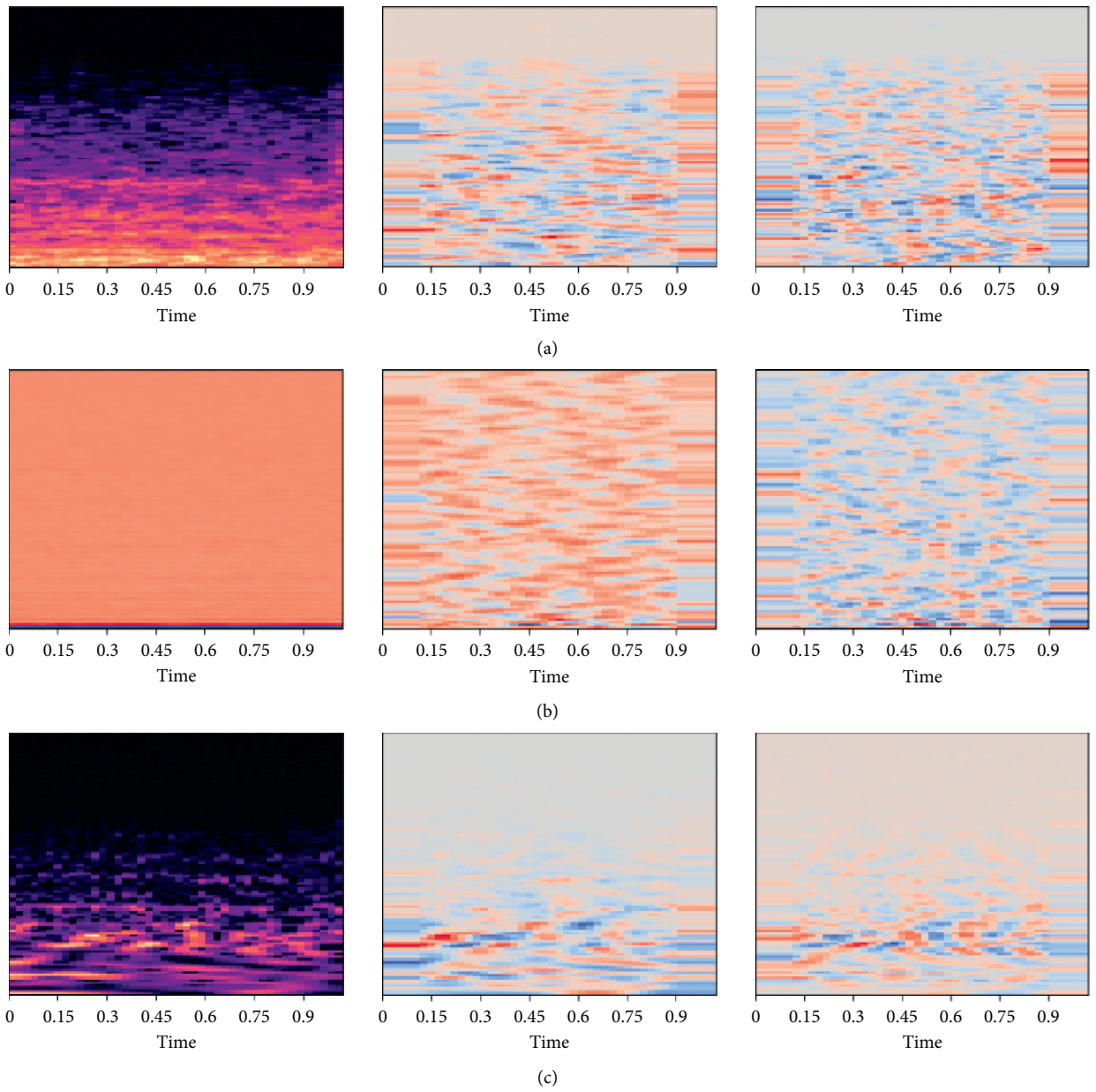

Figure 2: Images of extracted features, delta1 (Δ) and delta2 (Δ^2). (a) Mel spectrogram. (b) MFCCs. (c) CQT.

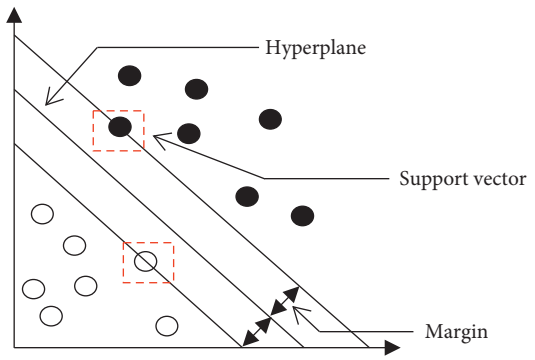

Figure 3: Example of two-class classification using SVM.

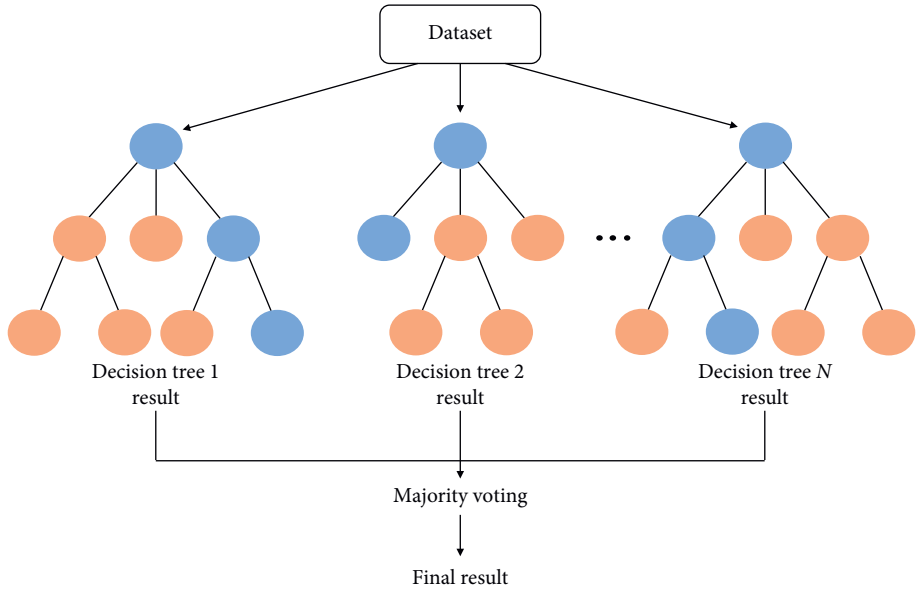

Figure 4: Random forest architecture.

classifier (decision tree) by adjusting the sample weight of the training data of the next classifier to continue learning.

$$\mathscr{L}^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t)}\right) + \sum_{i=1}^{t} \Omega\left(f_i\right), \tag{1}$$

$$\text{where } \Omega\left(f_i\right) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} \omega_j^2. \tag{2}$$

In equation (1), L represents an object function, l represents a loss function, and Ω represents a regularization term. The regularization term serves to prevent overfitting, while acting as a parameter to control the complexity of each classifier. It is represented by a linear combination of the γT term and L2 norm of the leaf weight term, which determine the number of branches by applying a penalty. A similar normalization technique is used in regularized greedy forest (RGF) [25]. More detailed information can be found in the work of Chen and Guestrin [26]. Figure 5 shows the general structure of XGBoost.

3.6. CNN. A CNN is an artificial neural network (ANN) model that is used primarily for image and video recognition, as well as image classification and analysis [27]. Existing ANNs are composed of only a fully connected layer. The input data must be flattened into one dimension. Therefore, there is a limit to learning, owing to the loss of spatial information. However, CNNs maintain the spatial information of the input data and effectively recognize features by adding convolutional and pooling layers. As shown in Figure 6, a CNN consists of the feature extraction and classification steps.

In the feature extraction process, image data are expressed as a matrix composed of three dimensions, and features are extracted through the convolutional and pooling layers. In the convolutional layer, a local pattern is learned through convolution operations that use filters to slide through the image. The data filtered through the convolutional layer are calculated using the activation function; moreover, nonlinearity can be applied, so that the model can reflect the nonlinear aspect of the data. In the pooling layer, the size of the input is reduced by a pooling rule that maps part of the input data to the value of the output data, thereby reducing the learning time and reinforcing the local pattern. The convolution and pooling processes are repeated several times in the CNN model, and a reduced-dimensional feature map, which is created through the feature extraction process, is used in the classification step. Overfitting problems often occur when the trained CNN model is suitable for the training data, but not the test data. To prevent overfitting, the model can be more generalized by applying a dropout process, where the nodes or units of the network are randomly deleted and trained.

In the classification step, through the aforementioned feature extraction step, multidimensional feature maps of the reduced size are arranged in one dimension. Thereafter, they pass through a fully connected layer that learns weights by connecting all the nodes. Finally, the classification is performed through an activation function.

3.6.1. Shallow CNN. In this study, we propose a shallow two-dimensional- (2D-) CNN model constructed based on the following basic modules: sequence of 2D-convolution, max-pooling, and dropout. The complete architecture involves a sequence of five basic modules, which is followed by batch normalization, global average pooling, and a dense layer with two outputs. The details are shown in Figure 7.

In the feature value extraction step, a feature map is created by sequentially passing the data through convolutional, pooling, and dropout layers. In the convolutional layers, a 3×3 and 3×1 2D-ConvFilter are used for

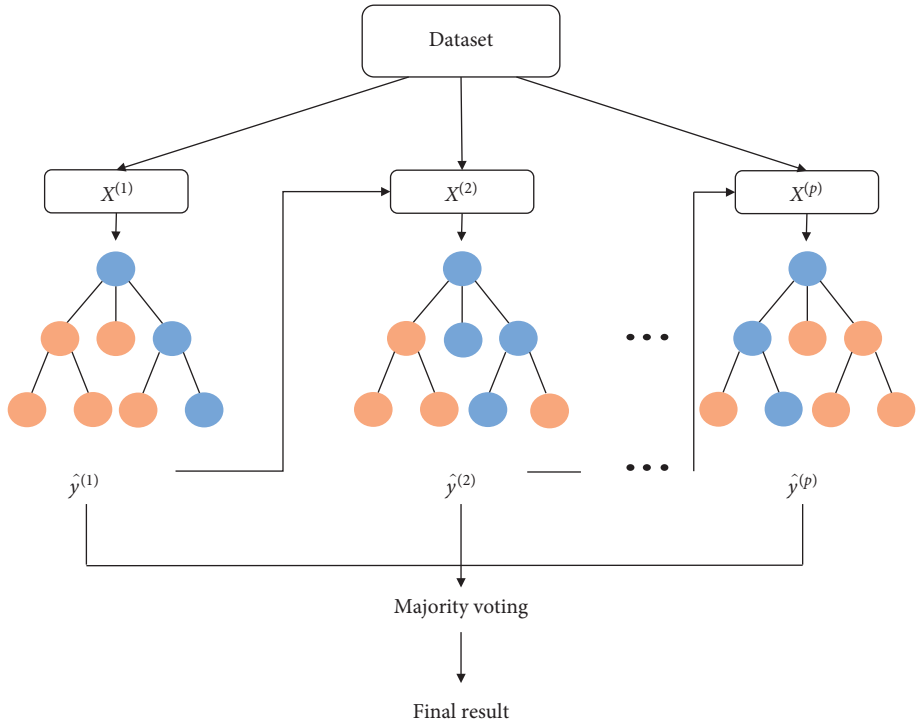

Figure 5: XGBoost architecture.

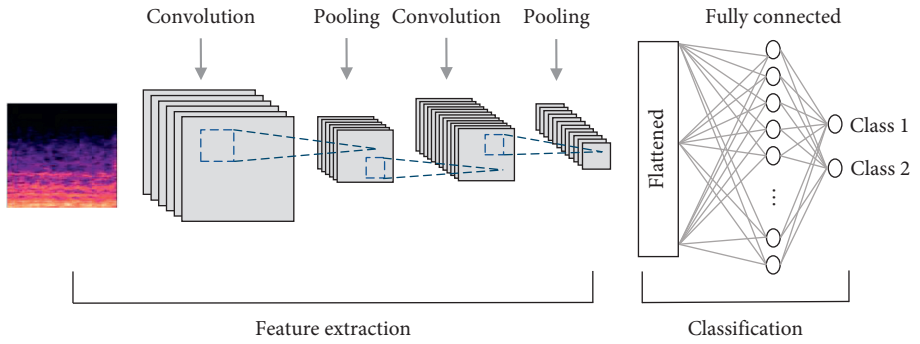

Figure 6: CNN architecture.

convolutional operations; 2×2 and 2×1 filters are sequentially used for the pooling operations. This is done to reduce the size of the entire feature map, while learning the local pattern according to time and frequency first. A 2×2 filter is then used for additionally learning features over time within a fixed frequency, after which the time interval is reduced again. In addition, a rectified linear unit (ReLU) function is used as the activation function of the convolutional layers. The pooling layer that receives the output data of the convolutional layer as the input data uses max-pooling, which is a method for extracting the maximum value in the pooling area. The ratio of the dropout that reduces the number of connections is 30%.

In the classification step, through a batch-normalization process, instead of the fully connected layer, a gap layer that can minimize the number of parameters and preserve the location information of the object is connected with a dense layer through an activation function. Finally, in the last output layer (dense layer), the classes are classified using the activation function, i.e., softmax.

3.6.2. VGGNet. VGGNet is a deep CNN model with an extremely low error rate. It was introduced in the 2014 ImageNet Large-Scale Visual Recognition Challenge [28]. It is easy to understand and has a simple structure. The important feature of VGGNet is the convolutional operation with a small filter size. Its nonlinearity increases when a 3×3 filter is used from the beginning to the end of the model compared to when the calculation is performed once using a large filter, which results in better performance. In addition, the number of parameters is smaller than that when a large filter is stacked. Nevertheless, with three fully connected and pooling layers, the number of parameters increases. This can result in problems, such as gradient vanishing and overfitting.

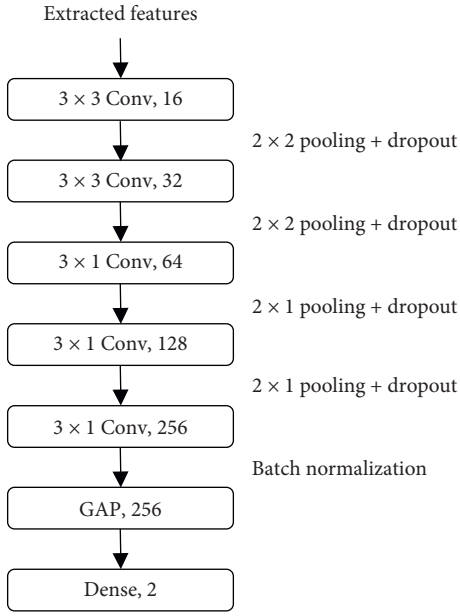

Figure 7: Overall structure of a shallow CNN. The numbers in each box, except for the output, represent the number of kernels. The number in the output box represents the number of classification classes.

Figure 8 shows the overall structure of the 13-layer VGGNet. The VGG-13 model comprises 2D convolutional, 2D max-pooling, dropout, and fully connected layers. The convolutional layer consists of filters with sizes of 64, 128, 256, and 512. The size of the feature map extracted through the convolutional operation using the padding technique is the same as that before the operation. Therefore, the size of the feature map is reduced only in the max-pooling layers, and the ratio of the dropout is 50%. The feature map created using 10 convolutional layers and 5 pooling layers is flattened to a one-dimensional vector form and reaches the output layer through 3 fully connected layers. In addition, the ReLU function is used as the activation function in the convolutional layers, and the softmax function is used in the last dense layer.

3.7. Grad-CAM. Grad-CAM is a technological approach that emphasizes the identification area for each class in the input image. It complements CAM, which is used only in CNNs. CAM has a global average pooling (GAP) layer [15]. It can be helpful in understanding the CNN's learning process and visualizing the internal representation. It is expressed by projecting a weighted feature map onto the original input image.

Grad-CAM is obtained using the following equations. In equation (3), u, v represent the width and height of the feature map, respectively; c represents the target class; y^c represents the activation function input; and A^k represents the kth feature map of the convolutional layer. When there are n feature maps of $u \times v$ spatial dimensions in the convolutional layer, each element of A^k is weighted according to the importance factor α_k^c. The weighted feature

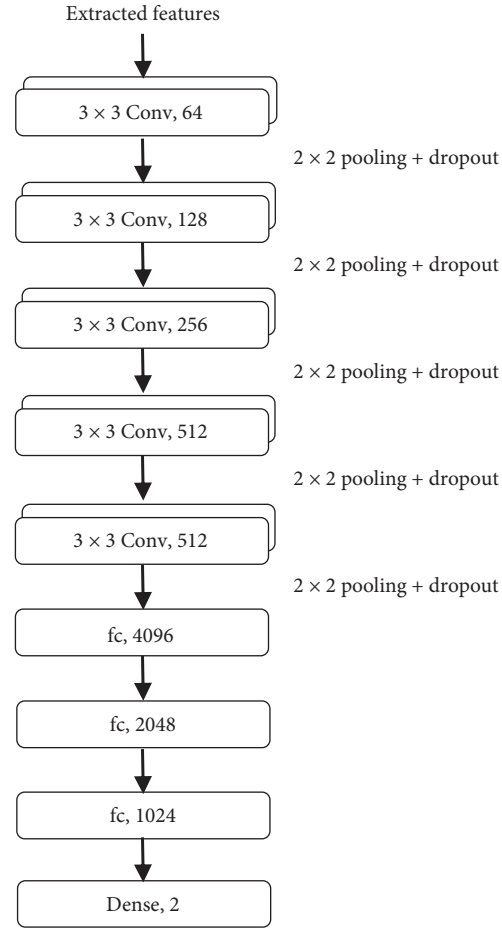

Figure 8: Overall structure of VGG-13. The numbers in the Conv boxes represent the number of kernels, and the numbers in the fc boxes represent the number of nodes. The last represents the number of classification classes.

maps activate only the features that have a positive effect on classification through the ReLU function. In equation (4), the importance factor α_k^c of the kth feature map is defined using the GAP method, which is a linear combination of $(\partial y^c / \partial A^k)$. This reflects the contribution of the feature map classified as class c.

$$L^c = \mathrm{ReLU}\left(\sum_k^n a_k^c A^k\right), \tag{3}$$

$$\alpha_k^c = \frac{1}{Z}\sum_{i,j}\frac{\partial y^c}{\partial A_{ih}^k}. \tag{4}$$

Because $L_{i,j}^c$ represents the importance of neuron activation in the spatial grid (i, j), L can be visualized using a heatmap to better show the identification area in the input images. The important areas in the heatmap are displayed in red. When the layer is deeper, the particle identification area of the heatmap is wider, making it easier to distinguish the layers visually. Figure 9 shows the process of Grad-CAM using the feature map generated during the training process of the CNN model and the gradients used to predict the target class.

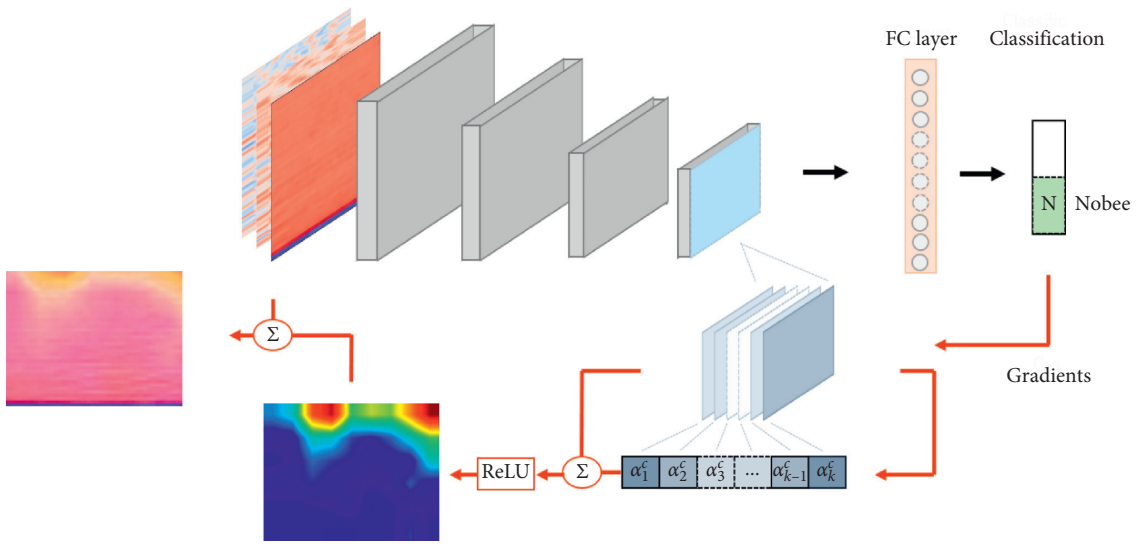

Figure 9: Process of Grad-CAM in CNN model.

4. Results

4.1. Experimental Setup. In this study, we conducted several experiments by applying features extracted from sound to the SVM, random forest, XGBoost, shallow CNN, and VGG-13 models. In the case of the SVM, random forest, and XGBoost models, the mean and standard deviation for each frame from the extracted features were used as input data. In addition, the CNN models used three-channel images as input data, wherein the extracted features and their corresponding delta1 and delta2 values were combined. The performance of the models was primarily compared using accuracy. Additionally, the recall, precision, and F1-score for each class were compared. For model validation, five-fold cross validation was used in all experiments.

Adjusting the parameters in machine learning algorithms significantly affects the performance of the model. Hence, to obtain optimal values, we used grid search for several key hyperparameters. Grid search is a method for obtaining optimal hyperparameters by generating a specific range of grids for the hyperparameters and applying the number of all cases to learning. The hyperparameters used in the search and their grid were set as follows: SVM: $C = \{0.01, 0.1, 1, 10, 100\}$, gamma = {0.01, 0.1, 1, 2, 5}; and random forest, XGBoost: number of estimators = {100, 200, 400, 800, 1200}, max depth = {4, 8, 12, 16, 20}, learning rate = {0.0001, 0.001, 0.01, 0.1, 0.2}, and min leaf samples = {3, 5, 7, 10, 15}. For the CNN models, hyperparameter search for the learning rate and batch size was conducted, and the Adam algorithm was used as the optimizer [29].

4.2. Classification Performance. Table 2 shows the classification accuracy for each feature using SVM, random forest, and XGBoost, along with the optimal hyperparameters selected using grid search. From the experimental results of each model, it can be observed that the classification performance is the best when MFCCs are used as input data. Moreover, it can be observed that, among the preprocessing methods, the MFCCs distinguish between bee sounds and noise better than the mel spectrogram and CQT. In addition, the optimal parameters for each extracted feature were selected differently; among the three models, XGBoost shows the highest accuracy (87.36%).

Table 3 shows the classification accuracy for each feature using the shallow CNN and VGG-13 models. For each model, the learning rates applied were 1e-5 and 1e-4, and the batch sizes used were 128 and 64. The classification accuracy of the shallow CNN model, using MFCCs with a learning rate of 1e-4 and a batch size of 128, was 88.42%, which was better than all the other accuracy values listed in Table 2. Furthermore, the VGG-13 model, using MFCCs with a learning rate of 1e-5 and a batch size of 64, showed the best accuracy (91.1%). Considering the results obtained through the feature extraction methods, the MFCCs have the best performance, followed by the mel spectrogram and CQT, which performed similarly.

Table 4 shows the precision, recall, and F1-score for the evaluation of the classification performance by class using MFCCs, which was the feature extraction method that demonstrated the best performance. When comparing the predictive performance for the Bee class, all models showed good results with an F1-score over 0.9. The difference in the performance between models was insignificant. However, when comparing the prediction performance for the noBee class, there was a significant difference. In the previous results, when comparing the accuracy of each model, the difference between the SVM and VGG-13 models, which showed the greatest difference, was as small as 6%. However, the F1-scores for the noBee class of the SVM and VGG-13 models were 0.44 and 0.79, respectively, which showed a difference of more than 0.3. Therefore, when considering

Table 2: Performance comparison of different features using various machine learning algorithms.

Model	Features	Parameters	Accuracy (%)
SVM	Mel spectrogram	$C = 0.01$, gamma = 0.1	77.58
	MFCCs	$C = 10$, gamma = 0.01	85.63
	CQT	$C = 0.1$, gamma = 0.1	77.55
Random Forest	Mel spectrogram	n_estimators = 100 max_depth = 6 min_samples_leaf = 3	82.14
	MFCCs	n_estimators = 100 max_depth = 8 min_samples_leaf = 5	86.82
	CQT	n_estimators = 200 max_depth = 12 min_samples_leaf = 3	74.07
XGBoost	Mel spectrogram	n_estimators = 200 learning_rate = 0.01 max_depth = 4	82.98
	MFCCs	n_estimators = 800 learning_rate = 0.01 max_depth = 8	87.36
	CQT	n_estimators = 800 learning_rate = 0.2 max_depth = 20	74.35

Table 3: Performance comparison of different features using various CNNs.

CNN	Features	Learning rate	Batch size	Accuracy (%)
Shallow CNN	Mel spectrogram	1e-4	128	79.82
		1e-5	64	79.94
		1e-5	128	79.30
	MFCCs	1e-4	128	88.42
		1e-5	64	88.04
		1e-5	128	87.89
	CQT	1e-4	128	80.70
		1e-5	64	79.12
		1e-5	128	79.47
VGG-13	Mel spectrogram	1e-4	128	84.91
		1e-5	64	80.58
		1e-5	128	91.10
	MFCCs	1e-4	128	90.88
		1e-5	64	91.93
		1e-5	128	91.75
	CQT	1e-4	128	85.09
		1e-5	64	80.88
		1e-5	128	80.35

both the classification accuracy and classification performance by class, it was confirmed that the performance of the VGG-13 model was the best.

4.3. Visualization of Features Using Grad-CAM. In this section, we provide the visualization results using Grad-CAM for validating the classification results of the CNN model. The features used as input data for the CNN model can be viewed as images, where the x-axis and y-axis represent time and frequency, respectively. The CNN model trained using these images creates an activation map during the learning process. Because the activation map is derived from the ground truth, it can be used to represent the input pattern learned in the CNN. Therefore, the performance of the model was verified through Grad-CAM visualization using the VGG-13 model with the MFCCs, which showed the best performance in the above experiments as input data.

Table 5 shows the annotation details of some files in the test set. Figure 10 shows the Grad-CAM visualization of the feature map created in the last convolutional layer of the VGG-13 model using those files. Each of the three pictures shows the original MFCCs, activation map, and the combination of the activation map and MFCCs. The area inside the red dotted box is considered an important pattern when classifying classes. Figures 10(a) and 10(c) show strong positive activation in all sections of the high-frequency area, and Figure 10(b) shows strong positives in a larger-frequency area. Therefore, it is recognized as an important pattern for predicting the Bee class. Conversely, the MFCCs that visualized the sound signals annotated as noBee showed strong positive activation in a specific section. The green dotted line indicates the start and end of the noBee section according to the annotation details in Table 5. When compared with the activated area, the section containing noise other than that of the bee and the activation map section show strong positive activation correlation. Consequently, we can observe the differences that humans cannot directly identify in the original MFCCs. Through annotation and comparison, it was possible to check whether the VGG-13 model performed classification effectively.

5. Discussion

This study aimed to verify the performance of a beehive sound classification model using various machine learning algorithms, by implementing a mel spectrogram, MFCCs,

Table 4: Detailed classification performance by class using MFCCs.

Model	Class	Precision	Recall	F1-score
SVM	Bee	0.99	0.85	0.92
	noBee	0.29	0.91	0.44
Random Forest	Bee	0.98	0.88	0.93
	noBee	0.47	0.88	0.61
XGBoost	Bee	0.98	0.90	0.94
	noBee	0.54	0.90	0.67
Shallow CNN	Bee	0.99	0.90	0.94
	noBee	0.56	0.93	0.70
VGG-13	Bee	0.99	0.94	0.96
	noBee	0.73	0.85	0.79

Table 5: Annotation details.

	Data annotation	Segment details
(a)	Bee	0–1 s Bee
(b)		0–1 s Bee
(c)		0–1 s Bee
(d)	noBee	0–0.31 s noBee
(e)		0.32–1 s noBee
(f)		0–1 s noBee

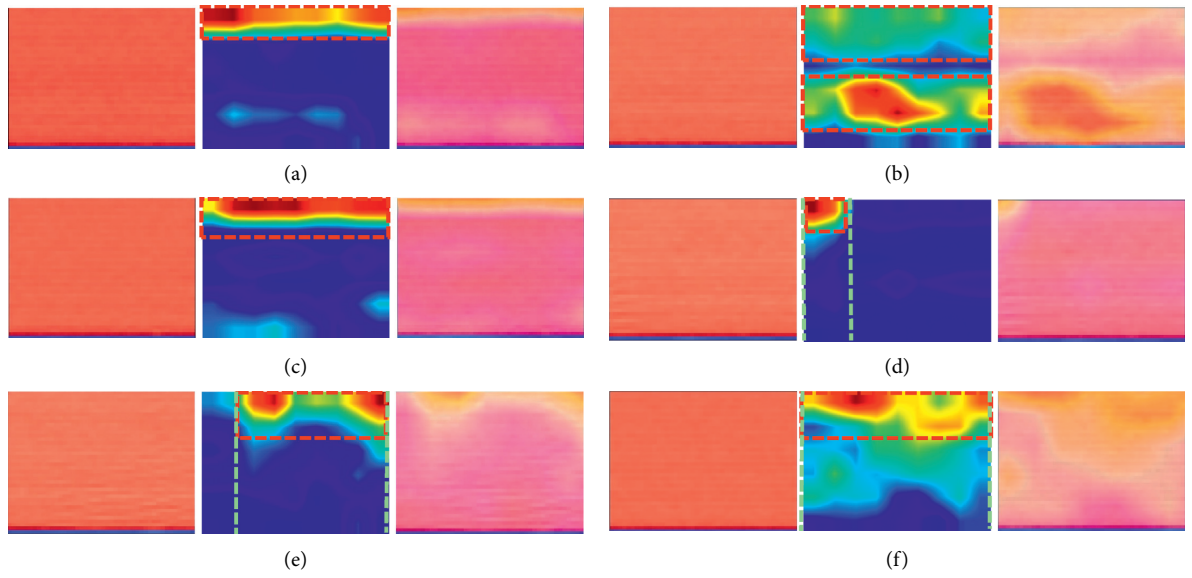

Figure 10: Visualization of MFCCs. Discriminative regions captured using VGG-13 are shown in heatmaps for the last convolutional layer. (a), (b), and (c) are annotated Bee; (d), (e), and (f) are annotated noBee.

and CQT, and to establish the optimal preprocessing method and classification model. We used five models: SVM, random forest, XGBoost, shallow CNN, and VGG-13. First, as a result of applying the machine learning algorithm that was used as input data, by extracting statistics after preprocessing, the models based on MFCCs performed best among SVM, random forest, and XGBoost. Of these, the best performing model was XGBoost, with an accuracy of 87.36%. Second, as a result of applying the shallow CNN and VGG-13 models, using the features in image form as input data, the models based on MFCCs preprocessing method also demonstrated the best performance. Furthermore, they showed better performance than the previous three models, which used statistics. Eventually, it was established that the CNN-based model was more useful in distinguishing sounds as Bees and noBees, while undergoing a short preprocessing operation.

In classifying beehive sounds, the noBee class contains various sounds. Hence, the risk factors that threaten bees may also be included. Furthermore, the noBee class is much

smaller than the Bee class, and the problem of detecting non-bee sounds is directly connected to whether the beehive is abnormal. Therefore, to verify whether the model predicts the noBee class effectively, we evaluated it using precision, recall, and F1-score for each class. The F1-score for the noBee class of the VGG-13 model was 0.79, demonstrating that the VGG-13 model was the most suitable for detecting non-bee sounds.

Based on previous experiments, we proposed a visualization method using Grad-CAM to verify how the VGG-13 model with the highest classification accuracy was learned. Grad-CAM was applied using the weight of the last convolutional layer and the gradients used to predict the target class. In the case of bee sounds, because strong activation occurs for all the time intervals of a specific frequency, all intervals are crucial in classifying the bee class. For non-bee sounds, specific sections containing other sounds were identified as important factors in predicting the noBee class. Finally, we established that the VGG-13 model could effectively determine the time interval in which the non-bee sound was involved.

In the previous study [14], the classification results were compared using the SVM and CNN models by dividing the beehive sound file into 60-second intervals and applying MFCCs and mel spectrograms. In the prediction of the test set, the AUC score of the SVM was 0.8, which was better than that of the CNN. However, in this study, the classification problem was addressed through more sensitive energy changes by dividing the audio file into shorter lengths. Because of attempting to establish the optimal model through various hyperparameter searches, the classification accuracy of the VGG-13 model reached 91.93%. In addition, by providing the visualization results for the VGG-13 model, we were able to verify the high classification performance of the CNN-based model.

6. Conclusion

In this study, we demonstrated the effectiveness of deep CNNs in classifying bee and nonbee sounds recorded within a beehive. Experiments using various machine learning algorithms and feature extraction methods on the sounds confirmed that the classification results of the VGG-13 model were the best. In addition, Grad-CAM was used to analyze how the VGG-13 model performed in sound scene classification, by visualizing sound signals and showing that the CNN-based model can distinguish between differences in the sounds that cannot be directly distinguished by humans.

We presented a CNN-based model and preprocessing method that were optimized for beehive sound data. The results demonstrated that sound analysis using a CNN model with an appropriate preprocessing method can effectively distinguish between bee and nonbee sounds. Therefore, abnormalities in the beehive can be identified early. This approach can help in building an automated monitoring system. In our future studies, we will attempt to obtain better results by applying multiple labels to sections where multiple sounds are heard simultaneously, thereby expanding this to a multiclass classification problem.

Data Availability

Beehive sound data were obtained from the Open Source Beehives (OSBH) project.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Authors' Contributions

Jaehoon Kim and Jeongkyu Oh contributed equally to this manuscript.

Acknowledgments

This study was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF), funded by the Ministry of Education (2017R1D1A3B03028084 and 2019R1I1A3A01057696).

References

[1] Food and Agriculture Organization of the United Nations, *Pollinators Vital to Our Food Supply under Threat*, Food and Agriculture Organization, Rome, Italy, 2020, http://www.fao.org/news/story/en/item/384726/icode/.

[2] J. D. Evans, C. Saegerman, C. Mullin et al., "Colony collapse disorder: a descriptive study," *PloS One*, vol. 4, Article ID e6481, 2009.

[3] L. U. Chensheng, K. M. Warchol, and R. A. Callahan, "Sublethal exposure to neonicotinoids impaired honey bees winterization before proceeding to colony collapse disorder," *Bulletin of Insectology*, vol. 67, no. 1, pp. 125–130, 2014.

[4] I. Kiskin, B. P. Orozco, T. Windebank et al., "Mosquito detection with neural networks: the buzz of deep learning," 2017, http://arxiv.org/abs/1705.05180.

[5] I. F. Rodriguez, R. Mégret, E. Acuna, J. L. Agosto-Rivera, and T. Giray, "Recognition of pollen-bearing bees from video using convolutional neural network," in *Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 314–322, IEEE, Lake Tahoe, NV, USA, March 2018.

[6] S. Adavanne, G. Parascandolo, P. Pertilä, T. Heittola, and T. Virtanen, "Sound event detection in multichannel audio using spatial and harmonic features," 2017, http://arxiv.org/abs/1706.02293.

[7] I. Y. Jeong, S. Lee, Y. Han, and K. Lee, "Audio event detection using multiple-input convolutional neural network," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events (DCASE)*, Munich, Germany, November 2017.

[8] S. Adavanne and T. Virtanen, "A report on sound event detection with different binaural features," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events (DCASE)*, Budapest, Hungary, March 2017.

[9] Y. Han, J. Park, and K. Lee, "Convolutional neural networks with binaural representations and background subtraction for acoustic scene classification," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop*, pp. 46–50, Munich, Germany, November 2017.

[10] G. K. Liu, "Evaluating gammatone frequency cepstral coefficients with neural networks for emotion recognition from speech," 2018, http://arxiv.org/abs/1806.09010.

[11] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *Proceedings of the 2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6, IEEE, Boston, MA, USA, September 2015.

[12] V. Boddapati, A. Petef, J. Rasmusson, and L. Lundberg, "Classifying environmental sounds using image recognition networks," *Procedia Computer Science*, vol. 112, pp. 2048–2056, 2017.

[13] Y. Su, K. Zhang, J. Wang, D. Zhou, and K. Madani, "Performance analysis of multiple aggregated acoustic features for environment sound classification," *Applied Acoustics*, vol. 158, Article ID 107050, 2020.

[14] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," 2015, https://arxiv.org/abs/1512.04150.

[15] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: visual explanations from deep networks via gradient-based localization," 2016, https://arxiv.org/abs/1610.02391.

[16] I. Nolasco, A. Terenzi, S. Cecchi, S. Orcioni, H. L. Bear, and E. Benetos, "Audio-based identification of beehive states," in *Proceedings of the ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8256–8260, IEEE, Brighton, UK, May 2019.

[17] I. Nolasco and E. Benetos, "To bee or not to bee: investigating machine learning approaches for beehive sound recognition," 2018, https://arxiv.org/abs/1811.06016.

[18] "Open source beehives project," 2020, https://www.osbeehives.com/.

[19] "LIBROSA," 2015, https://librosa.github.io/librosa/.

[20] J. Shen, R. Pang, R. J. Weiss et al., "Natural tts synthesis by conditioning wavenet on mel spectrogram predictions," in *Proceedings of the 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4779–4783, IEEE, Calgary, Canada, April 2018.

[21] M. Deng, T. Meng, J. Cao, S. Wang, J. Zhang, and H. Fan, "Heart sound classification based on improved MFCC features and convolutional recurrent neural networks," *Neural Networks*, vol. 130, pp. 22–32, 2020.

[22] T. Lidy and A. Schindler, "CQT-based convolutional neural networks for audio scene classification," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2016 Workshop (DCASE2016)*, pp. 1032–1048, IEEE Budapest, Budapest, Hungary, September 2016.

[23] B. M. Whitaker, P. B. Suresha, C. Liu, G. D. Clifford, and D. V. Anderson, "Combining sparse coding and time-domain features for heart sound classification," *Physiological Measurement*, vol. 38, no. 8, pp. 1701–1713, 2017.

[24] T. Lay, T. H. Dat, and B. Ma, "An integrated solution for snoring sound classification using Bhattacharyya distance based GMM supervectors with SVM, feature selection with random forest and spectrogram with CNN," in *Proceedings of the Conference of the International Speech Communication Association*, Stockholm, Sweden, August 2017.

[25] R. Johnson and T. Zhang, "Learning nonlinear functions using regularized greedy forest," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 5, pp. 942–954, 2013.

[26] T. Chen and C. X. Guestrin, "A scalable tree boosting system," in *Proceedings of the 22nd Acm Sigkdd International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, San Francisco, CA, USA, August 2016.

[27] A. Khamparia, D. Gupta, N. G. Nguyen, A. Khanna, B. Pandey, and P. Tiwari, "Sound classification using convolutional neural network and tensor deep stacking network," *IEEE Access*, vol. 7, pp. 7717–7727, 2019.

[28] C. Gousseau, "VGG CNN for urban sound tagging. DCASE2019 Challenge," *Technical Report*, 2019.

[29] S. Bock and M. Weiß, "A proof of local convergence for the Adam optimizer," in *Proceedings of the 2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, IEEE, Budapest, Hungary, July 2019.