Article

An Ensemble One Dimensional Convolutional Neural Network with Bayesian Optimization for Environmental Sound Classification

Mohammed Gamal Ragab ¹ , Said Jadid Abdulkadir ^{1,2,*} , Norshakirah Aziz ^{1,2}, Hitham Alhussian ^{1,2}, Abubakar Bala ^{3,4} and Alawi Alqushaibi ¹

- ¹ Department of Computer and Information Sciences, Universiti Teknologi PETRONAS, Seri Iskandar 32610, Malaysia; mohd.gamal_20497@utp.edu.my (M.G.R.); norshakirah.aziz@utp.edu.my (N.A.); seddig.alhussian@utp.edu.my (H.A.); alawi_18000555@utp.edu.my (A.A.)
- ² Centre for Research in Data Science (CERDAS), Universiti Teknologi PETRONAS, Seri Iskandar 32610, Malaysia
- ³ Electrical and Electronics Engineering Department, Universiti Teknologi PETRONAS, Seri Iskandar 32610, Malaysia; abubakar_15002355@utp.edu.my
- ⁴ Electrical Engineering Department, Bayero University Kano, Kano 700241, Nigeria
- * Correspondence: saidjadid.a@utp.edu.my

Citation: Ragab, M.G.; Abdulkadir, S.J.; Aziz, N.; Alhussian, H.; Bala, A.; Alqushaibi, A. An Ensemble One Dimensional Convolutional Neural Network with Bayesian Optimization for Environmental Sound Classification. *Appl. Sci.* **2021**, *11*, 4660. <https://doi.org/10.3390/app11104660>

Academic Editor: Jerome Antoni

Received: 1 April 2021

Accepted: 29 April 2021

Published: 19 May 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Abstract: With the growth of deep learning in various classification problems, many researchers have used deep learning methods in environmental sound classification tasks. This paper introduces an end-to-end method for environmental sound classification based on a one-dimensional convolution neural network with Bayesian optimization and ensemble learning, which directly learns features representation from the audio signal. Several convolutional layers were used to capture the signal and learn various filters relevant to the classification problem. Our proposed method can deal with any audio signal length, as a sliding window divides the signal into overlapped frames. Bayesian optimization accomplished hyperparameter selection and model evaluation with cross-validation. Multiple models with different settings have been developed based on Bayesian optimization to ensure network convergence in both convex and non-convex optimization. An UrbanSound8K dataset was evaluated for the performance of the proposed end-to-end model. The experimental results achieved a classification accuracy of 94.46%, which is 5% higher than existing end-to-end approaches with fewer trainable parameters. Four measurement indices, namely: sensitivity, specificity, accuracy, precision, recall, F-measure, area under ROC curve, and the area under the precision-recall curve were used to measure the model performance. The proposed approach outperformed state-of-the-art end-to-end approaches that use hand-crafted features as input in selected measurement indices and time complexity.

Keywords: Bayesian optimization; convolutional neural networks; deep learning; ensemble learning; environmental sound classification; optimization; UrbanSound8k

1. Introduction

Environmental sound classification is a sound recognition system that identifies sound events in the real world, identified as sound event recognition. In recent years, sound problems have received noticeable attention, with popular applications from crime detection [1] to environmental context-aware processing [1], healthcare [2], recognition of automatic speech [3], music information [4], noise mitigation [5], music classification [6], and smart audio-based surveillance systems [7,8]. Most environmental classification approaches depend on hand-crafted features such as typical automatic classification systems or mid-level representations [6], which obtain a good trade-off between model accuracy and its computational cost [9–22]. Environmental sound taxonomy usually has two components: acoustic

and classification characteristics. The environmental sound classification is intended to identify the type of event representing a specific sound [23].

For an urban sound event, several typical sound characteristics like mel-frequency coefficients [24], zero-crossing [25], and wavelet transformation [26] have been used as sound features representation. Many classifiers are commonly applied in sound-related classification problems, for instance, support vector machines, extreme learning machines, and Gaussian mixture, in addition to standard machine learning algorithms such as k-nearest neighbors [27–30]. Nevertheless, traditional techniques are designed to model fewer variations, leading to a lack of frequency and time [29]. The neural network has been the primary choice for environmental recognition, and has been superior to conventional classifiers in the last few years [13].

It proved its efficiency to solve complex categorizing problems compared to traditional classifiers. Convolution neural networks (CNNs) are considered as an appropriate technique for signal tasks due to their ability to capture time and frequency features [31], and a suitable choice for spectrogram-like inputs data [32]. CNNs architecture has three types, which are 1D, 2D, and 3D CNN [33]. The 1D CNN is often used for sequence data processing [34] and was designed to process data in multiple signals such as audio spectrogram [35,36], 2D CNN is mainly used for image and text recognition [35], and 3D CNN is essentially applied to video recognition and medical applications [37].

Recently, CNN has been applied to the analysis sensor, time series, and periodical data [34,38,39]. It has had a huge influence on many audio and music processing tasks using sensor data, periodic signal analysis [34], and diverse acoustic environments data [40,41]. Furthermore, it has significant outcomes on several tasks in the various fields such as audio classification of video clips [42], genre classification [6], music tagging [43], speech identification [44], large-scale video clip classification [42], speaker identification [44], and environmental sound classification [32,45–49]. Although using 1D CNNs in environmental sound classification tasks has substantially improved, they still have a long way compared to 2D CNN based image classification algorithms. 1D CNN is similar to a traditional neural network; however, instead of hand-crafted features, it usually takes the raw signal data [34]. Then, input data is processed through several convolution layers to learn both (low and high) representation levels.

Three different approaches have been proposed to apply CNNs on sound event recognition: (i) logmel-CNN [45], where CNN is used as a classifier, and log-mel features as the input; (ii) raw-CNN [50], where CNN extracts audio features from raw waveforms and classifies them without the need for feature engineering; and (iii) hybrid-CNN [49], which uses an average fusion method to combine both raw-CNN and logmel-CNN approaches. Although substantial improvements have been made in the above approaches, there are still some drawbacks, such as: (i) some audio event information that may be critical may not be captured; (ii) in terms of classification performance and model generalization ability, the performance of traditional recognition algorithms is very restricted; (iii) hybrid models have the disadvantage of neglecting the trend that leads to recognition errors of individual models; and (iv) end-to-end approaches cannot exceed 2D CNN in terms of model generalization and recognition accuracy with feature learning.

To overcome the mentioned limitations, in this paper, we propose a new one-dimensional end-to-end CNN integrated with Bayesian optimization (BO) [51] and ensemble learning [23,52] methods that learn directly from the audio signal. It also offers a consolidated design to reduce the computational complexity and the needed data for model training, aiming to improve model prediction, generalization, and robustness over a single classifier. Comprehensive and comparative experiments are conducted using UrbanSound8k [53] dataset to evaluate the proposed method performance. The experiment results indicated that our model recognition achieves state-of-the-art performance, outperforms traditional methods, and achieved higher accuracy up to 94.46% which is 5% higher than existing end-to-end approaches. The proposed approach has few trainable parameters compared to 2D CNNs, which have millions of parameters [32,45,54].

2. Related Work

The research community has paid more attention to the environmental sound recognition problems with the popularity of CNNs, and its application that exceeds conventional methods for different categorization problems [36,45]. Deep learning [34,35] is a particular technique of machine learning that is characterized by its hierarchical multi-level architecture with subsequent phases in data processing [35]. It can extract and learn features directly from raw waveform signals that have efficiently proven several problems in language and images recognition [55,56]. CNNs are widely applied as a modeling algorithm for multi-variable time-series input data due to the challenges in hand-crafted feature-based methods [32,45,46,49,57]. However, most of these methods require the input data to be transformed into a 2D representation and feed the input to CNNs architecture as AlexNet and VGG, which were developed initially for object recognition problems [58–60].

According to many studies, CNNs have fewer parameters to optimize than other unidirectional, and deep neural networks [61]. Deep learning methods, particularly CNNs, are easier to train, unlike other traditional methods, since they provide a structure that performs features extraction and classification in one single block instead of hand-crafted features and classification separately. Many applications consider the log-mel feature of audio signals as an effective audio recognition function. It calculates each sound frame and its magnitude of each frequency area. Each frame's features can be arranged alongside to time axis to create a features map [62]. That makes convolution layers in CNNs to be used as a classifier with the log-mel feature map. This method was first applied and proposed by Piczak et al. (2015) in the environmental sound classification [45]. They attempted to get the delta-log-mel and log-mel features extracted from each frame and then used the delta and two-dimensional static log-mel features map, and then they used two-channel CNN for classification.

Similarly, Bello (2017) [32] used a two-channel network classification technique with log-mel features map. They modified the network structure that consists of three convolution layers and one fully-connected layer. Besides, they utilized data augmentation techniques to increase training data variation in order to train the network efficiently, which enhanced the classification accuracy by 6%. In the same year, Dai et al. (2017) [63] introduced various deep convolutional models that consist of residual learning and utilize down-sampling and batch normalization technique in the CNN initial layers, which achieved an accuracy of 72% on the UrbanSound8k dataset.

On the one hand, many researchers attempt to automatically learn features from raw waveforms [64,65]. However, that could not outperform the classification accuracy obtained by log-mel features [49]. For instance, Boddapati et al. (2017) [58] experiments achieved an average accuracy of 92.5% by using mel-frequency, spectrogram, and cross-recurrence alongside AlexNet and GoogLeNet as classifiers with 2D input representations. Its efficiency in summarizing waveforms with high-dimensional spectrograms into a compact representation is one of the primary benefits of using 2D representations. However, they rely on large amounts of the dataset to learn without overfitting, making sound classification modeling difficult using two-dimensional CNNs [47].

1D CNNs have been widely used in major engineering applications, especially focused on signal processing. Their state-of-the-art performance is highlighted and verified with their unique properties [34]. There are still limited studies in end-to-end environmental research. Hoshen et al. (2015) [50] found that the time difference between channels has been found to indicate the input position based on the creation of an end-to-end multi-channel 1D CNN for audio recognition with an error rate or 27.1% for single-channel. They stated that direct learning from the sound waveform matches the log-mel filter-bank performance. Tokozume et al. (2017) [49] developed a method named between-class learning that mixture of two audio samples then trains neural networks to anticipate the mixing ratio of these samples. That method has improved efficiency for different architectures used for sound recognition tasks according to their experiments. Besides, they developed an end-to-end

method that outperforms traditional learning techniques on various datasets using the between-class learning approach and 1D CNN.

End-to-end applications using 1D CNNs are becoming common in signal processing because of their ability to benefit from its structure that can learn from waveforms directly [34,50]. An end-to-end speech recognition approach based on waveforms and multi-scale convolutions that directly learn input data representation from signal was proposed by Zhu et al. (2017) [66]. They used several 1D convolution layers with different kernels to extract data features, and then to ensure consistent sampling frequency, they used a pooling layer to concatenate features. Their experiments reported an error rate of 23.28%. Ravanelli and Bengio developed another end-to-end technique for speech identification (2018) [44]. They tried to learn the (low and high) cut-off data frequencies of filters. Their model reduced the proposed model parameters by learning practical filters from the first layer. They achieved 0.85% as a sentence error rate based on the TIMIT dataset. In the same year, Zeghidour et al. [67] developed an end-to-end architecture using a 1D CNN with filter-banks without using mel-filter for speech recognition. Furthermore, Abdoli et al. (2019) [47] proposed a sound classification method based on the 1D CNN. They used a gamma-tone filter bank for features generation. Their method was tested on the UrbanSound8k dataset with a classification accuracy of 89%.

It was suggested that the hybrid model integrating two or more models could achieve high prediction accuracy [17]. A robust prediction can be achieved by combining several CNN networks that learn from waveform raw audio as one or two representations of the signal. Li et al. (2018) [23] combined two different networks; one learns the audio waveform directly, and the second using log-mel features to learn high-level representations from the waveform. Both models are independently trained, and the two model predictions are combined with the Dempster–Shafer method. An accuracy of 92.2% was achieved using this ensemble method on UrbanSound8k datasets with a 2% improvement in recognition accuracy compared to the current end-to-end model.

3. Methodology

Deep learning neural networks offer increased flexibility and scalability in a proportion of data availability. However, a downside to this versatility is that they are sensitive to the training data details and can find a new set of weights that generate different predictions from the learning procedure through a stochastic algorithm. We refer to this as high variance neural networks [35]. Creating several models rather than a single model, and combining them is a practical approach to reducing neural networks' variance. This approach decreases the uncertainty of predictions and can lead to a more stable and sometimes better prediction than each member model's predictions. CNNs substantially progressed in numerous recognition problems by replacing manually engineered features techniques. In a sequence of layers, the model is created, starting with convolutional and pooling layers. Units are organized into features map in the convolutional layers and linked to local patches in the previous layers features map. A nonlinearity activation function is then transferred via an activation function such as ReLU [39]. In this way, within local groups of values, CNN defines data correlated and invariant to the location. Several convolutions, pooling layers, and nonlinearity activation functions are stacked after each other, followed by dense layers. Finally, a fully-connected layer with softmax function as output classification layer.

The proposed network was developed and built to optimize network parameters and map the input data according to the convolutional layers hierarchical feature extraction mechanism. To increase the region wrapped by the next receptive fields, we stacked different pooling layers after each convolutional and before batch normalization layers. The final convolution layer output is then flattened and used as the input of several fully linked stacked layers. The primary drawback of using 1D CNNs is that the input samples must be in a fixed length. However, different durations of the sound captured from the environment can occur; splitting the audio signal into multiple fixed-length frames using

the sliding window of acceptable width is one way to overcome this restriction. Therefore, we applied a variable width window in our approach to conditionalize the signal of each audio to the proposed model input layer. The width of the window mainly relies on the signal samples rate. Sequential audio frames can similarly have some overlap to optimize the use of data. Based on Abdoli et al. (2019) [47], a sample rate of 16 kHz or 18 is considered an acceptable compromise between the input sample quality and model computational complexity. Our proposed architecture aims to manage variable duration audio, and directly learn from the audio waveform with distinctive representation that will obtain an acceptable classification performance on various environmental sounds.

3.1. Ensemble Learning

Ensemble learning has become a hot subject recently. Many studies have demonstrated that ensemble learning performance is superior to a single classifier [68,69]. Ensemble machine learning algorithms combine several learning model predictions into a single “ensemble” model to maximize their efficiency [68]. Predictions that are good in various ways will result in a more stable and often better prediction than any individual member. Bagging, boosting, and stacking are traditional approaches to ensemble learning [70]. Since the classification method of ensemble learning often obtains better accuracy of classification than an individual classifier. However, the enhancement comes at the cost of computational time [71]. In essence, ensemble learning is proposed to decrease the randomness of the outcome of a one-time prediction. Multiple predictors can be obtained in two different primary ways, which are: (i) by using various algorithms to process the same dataset, and (ii) using various machine learning algorithms with a change of its hyperparameters. In general, changing algorithms’ hyperparameters is crucial for optimization and is not related to integration. Therefore ensemble prediction models can be divided into two types; namely, homogeneous and heterogeneous ensemble models [72], which correspond to (i) and (ii), respectively.

Training each model on various training data folds is one way of achieving differences between models. Models are naturally trained on various subsets of training data using resampling methods, such as cross-validation and bootstrap [73]. For a deep neural network model, learning the weights involves solving a high-dimensional problem of non-convex optimization [74]. The difficulty in solving this problem of optimization is that there are many “good” solutions, and the learning algorithm can bounce around and settle in one of them [75]. This is referred to as convergence with a stochastic optimization solution, where a collection of unique weight values defines a solution. A type of heterogeneous ensemble strategy is the stacking of ensemble learning. The heterogeneous ensemble combines many different base classifiers into a robust meta-classifier to increase the strong classifier’s generalization potential. The ensemble approach takes advantage of both the base classifiers and the meta-classifier learning abilities, significantly improving the accuracy of the classification and the correctness rate, as shown in Algorithm 1.

Algorithm 1 Ensemble Learning Procedure

```

1: Input: Training data  $D = \{x_i, y_i\}_{i=1}^m$ 
2: Output: ensemble meta-classifier  $H$ 
3: Step 1: learn baseline classifiers
4: For  $c = 1$  to  $n$  Do:
5:   Learn  $h_t$  based on  $D$ 
6: End For
7: Step 2: build up new dataset for prediction
8: For  $i = 1$  to  $m$  Do:
9:    $D_h = \{x'_i, y_i\}$ , where  $x'_i = \{h_1(x_i), \dots, h_T(x_i)\}$ 
10: End For
11: Step 3: learn meta-classifier
12: Return  $H$ 

```

3.2. Proposed Architecture

Ensemble models can achieve lower generalization problems than individual models, but they are difficult to construct with deep neural networks because of the computational cost of training each model. Alternatively, multiple model snapshots should be trained during a single training, and predictions should be combined to create the ensemble classifier. However, this method has some drawbacks, such as the saved models being identical, leading to similar prediction errors and predictions that do not have many benefits from combining their predictions. The use of an aggressive learning rate forces significant changes in the model weights, and the essence of the model saved at each snapshot is one approach to promoting various models saved during a single training run. Several callbacks were used at some frequencies to monitor and track models during training. We have used checkpointing to save the network weights only when the validation dataset increases the classification accuracy. It enables us to monitor the model at the end of the epoch or batch during training. Pseudocode of the proposed method shown in Algorithm 2.

Algorithm 2 A Pseudo-code of the Proposed Approach

```

1: Input: Dataset:  $X_{i=1}^{ms}$ , Bayesian learners
2: Output: ensemble meta-classifier
3: For  $i = 1$  to  $max\ iterations$  Do:
4:     Split dataset into  $k$ -folds
5:     For Each fold in  $l$ -folds Do:
6:         For Each predictor in ensemble Do:
7:             Train learner on train set in fold
8:             Validate class probabilities from learner in fold
9:             Create prediction matrix of classes probabilities
10:        End For
11:    End For
12:    Calculate probabilities across learners
13:    Get loss value of loss function with probabilities
14:    If  $loss < previous$  Then:
15:        Stack learner
16:    Else:
17:        Save model checkpoint
18:        Break For
19: End For

```

4. Experiments

This section includes an overview of the experiment's environment, data preprocessing, and hyperparameters configurations. The proposed architecture aims to manage variable duration audio signals, and directly learns from the audio waveform with a strong classification performance on various environmental sounds. There are six kinds of layers in the proposed model: (i) input layer; (ii) convolution layers for features extraction from input data; (iii) max-pooling layers for reducing dimensionality and enhancing the robustness of some features; (iv) flatten layer to convert features map to a single array; (v) fully-connected layer to integrate extracted features; and (vi) categorical softmax function as an output layer to represent distribution over different classes. Besides, we divided the dataset into a 5-cv following the technique used in [47], and a single training fold was used as a validation set for tuning the hyperparameter in Bayesian models. We implemented our network using TensorFlow/Keras [76], taking advantage of its native support for asynchronous execution and flexibility.

4.1. Feature Extraction

Several studies have shown that aggregate characteristics achieve higher environmental sound classification accuracy than single speech recognition characteristics [77]. Previous studies have found that the techniques of features extractions such as log-mel, mel-frequency cepstral coefficient, and mel spectrogram are considered to be the most suit-

able and frequently used auditory features in sound recognition [40,78]. They can capture different audio data patterns. Thus, network recognition based on these input methods can further use complementary relationships to enhance recognition efficiency further. All feature sets are combined linearly, alongside their time-frequency representations [41]. We used the same feature combination in our work. Librosa [79] package was used for features extraction with (60) bands to cover the frequency range of the sound segments and represent each segment feature as (frequency $\times$ time $\times$ channel).

4.2. Dataset Furthermore, Preprocessing

A large amount of training data is needed in multi-class sound classification problems, particularly with a high feature vector dimension. This paper dataset comes from Urbansound8k [53] and includes ten classes of 8732 urban sounds obtained from the real world (the duration is less than or equal to 4 s), totalling 9.7 h. The dataset is divided into ten groups of audio events: street music (SM), jackhammer (JH), siren (SI), drilling (DR), engine idling (EI), dog bark (DB), air conditioner (AC), siren (SI), car-horn (CH), children playing (CP), and gunshot (GS), with 126 male speakers and 125 female speakers as in shown in Table 1 and Figure 1.

Table 1. UrbanSound8k Dataset composition.

Class	Files	Length (s)
Street Music (SM)	1000	4000
Jackhammer (JH)	1000	3610
Drilling (D)	1000	3548
Engine Idling (EI)	1000	3935
Dog Bark (DB)	1000	3148
Children Playing (CP)	1000	3961
Air Conditioner (AC)	1000	3994
Siren (SI)	929	3632
Car Horn (CH)	429	1053
Gun Shot (GS)	374	616

Figure 1. UrbanSound8k dataset distribution boxplot.

All sound clips with a frequency of 22,050 Hz are converted to single-channel wave files. Similar to the augmentation procedure in [80], we put the window size to 1024 (approximately 47 ms), which is half of the window size as the hop-size and divide it into 50% overlapping segments. Data preprocessing techniques include normalization, splitting, and transformation. We used a stratified split category strategy to ensure fair distribution

between the various sections of the categories. The dataset randomly divided into five equivalent size subsets. We split the dataset into k -fold cross-validation ($k = 5$) and use a 20% of training folds for validation and tuning hyperparameter. We have carried out cross-validation of our networks five times. Our approach's data splitting and transformation procedure are provided in Equation (1).

$$\begin{bmatrix} \delta_1^1 & \delta_1^2 & \cdots & \delta_1^n \\ \delta_2^1 & \delta_2^2 & \cdots & \delta_2^n \\ \delta_3^1 & \delta_3^2 & \cdots & \delta_3^n \\ \vdots & \vdots & \ddots & \vdots \\ \delta_m^1 & \delta_m^2 & \cdots & \delta_m^n \end{bmatrix} \rightarrow \begin{bmatrix} \zeta_1^1 & \zeta_1^2 & \cdots & \zeta_1^n \\ \zeta_2^1 & \zeta_2^2 & \cdots & \zeta_2^n \\ \zeta_3^1 & \zeta_3^2 & \cdots & \zeta_3^n \\ \vdots & \vdots & \ddots & \vdots \\ \zeta_m^1 & \zeta_m^2 & \cdots & \zeta_m^n \end{bmatrix} \quad (1)$$

where δ_n^m represents the instance of 1d signal features, ζ_n^m is denoting to the classes output, $1 \leq i \leq m, 1 \leq j \leq n, m$ symbolizes the size of the dataset, and n is the size of input features.

4.3. Hyperparameter Tuning

Tuning hyperparameters involves the procedure that determines the network configurations' values, contributing to providing more accurate classification accuracy [81]. It highly influences machine learning algorithms. Tuning hyperparameters are typically done manually (trial and error). Nevertheless, this approach takes time to test all the possible hyperparameters that may result in an acceptable performance. Other approaches, such as grid searches [82,83], genetic algorithm [84], and particle swarm optimization [85,86], are widely used as hyperparameters tuning methods. Grid and random search divide the hyperparameter values into several folds to build a certain space range of grid search space. Then, it crosses over all grid points to find the best hyperparameters values. It requires that many hyperparameters be tested routinely by automatically re-training the model for each hyperparameter value [82], which makes it useful by mapping the problem space and giving more optimization possibilities. However, these methods and metaheuristic algorithms are restricted to a large-scale network. They also take time because experimental methods require several experiments to estimate the value.

Many researchers are attracted to BO as an efficient method for tuning networks hyperparameters [87]. Using a small number of samples can obtain the values of the optimum hyperparameters for optimization, unlike traditional optimization methods [88]. It does not need the functions of explicit expression [89]. BO is a very effective in solving this kind of optimization problem [87,90]. Nonetheless, for hyperparameter tuning in deep neural networks, the time required to evaluate the validation error for even a few hyperparameter settings remains a bottleneck. BO is slightly more sophisticated compared to manual tuning since it combines prior information about the unknown function with sample information to obtain posterior information of the function distribution. Then, based on this posterior information, we can deduce where the function obtains the optimal value [91]. In other words, It can be useful to model-based approaches for automatically configuring hyperparameters to generate a surrogate model of some unknown function that would otherwise be too expensive. In BO, rather than considering the objective function as a black-box about which we can only obtain point-wise information, regularity assumptions are made. These are used to actively learn a model of the objective function. The resulting algorithms are practical and provably find the global optimum of the objective function while evaluating the function at only few parameters [92].

These hyperparameters include the number of filters, kernels, optimizer, momentum, learning rate, batch size, and dropout rate. In experiments, the initial configurations of hyperparameters are randomly generated with shuffled training and validation data. Bayesian optimization searches for the best hyperparameters and fitting a new model of the best k -cv selection. For each point in the search space, Bayesian optimization cross-validation provides cross-validation estimates of performance statistics. Different data split between folds may generate different optimal tuning parameters. The tuning

hyperparameters were then selected with the lowest average cross-validation error. We refer to it as the optimal cross-validation choice for tuning hyperparameters. A detailed list of hyperparameters and search bounds are shown in Table 2. For the neural network training, we trained it through a variant of adaptive optimizers like Adam, RMSprop, AdaBound, and EAG [93] with initial learning rate of 10^{-4} , $\beta_1 = 0.90$ and $\beta_2 = 0.99$, $filter = 128$, and $kernal = 2$. We used an initial value of 0.25 as dropout probability in the ninth layer to prevent network overfitting at the training stage. The batch size is set to a value in the range of (4 and 128), while all weight parameters are subjected to L2 regularization. When analyzing a certain tested hyperparameters influence, the remaining parameters are fixed with default values. The detailed default model is shown in Table 3.

Table 2. Hyperparameters search space.

Parameters	Search Space
Number of filter	{16:512}
Kernel size	{2, 4, 6:12}
Batch size	{16:256}
Activation function	{ReLU, Linear, Sigmoid, Tanh}
Optimization method	{Adam, EAG, RMSprop, Nadam, Adamax, Adadelata, Adagrad}
Dropout rate	{0:0.5}

Table 3. Proposed model detailed parameters.

Layer	Type	Kernel $\times$ Filter	Other Parameters
1	Convo1D	2×128	Activation = ReLU, Strides = 2
2	MaxPool1D	—	Size = 2, Strides = 2
3	Convo1D	2×128	Activation = ReLU, Strides = 2
4	MaxPool1D	—	Size = 2, Strides = 2
5	Convo1D	2×128	Activation = ReLU, Strides = 2
6	Convo1D	2×128	Activation = ReLU, Strides = 2
7	Flatten	—	—
8	Dense	1×128	Activation = ReLU
9	Dropout	—	Rate = 0.25
10	Dense (Output)	$\{1 \times 10\}$	Activation = Softmax

4.4. Training Procedure

We selected top five models, indicated by $M = \{M_0, M_1, \dots, M_n\}$. M represents the top five models selected based on Bayesian optimization. The main difference between them is the optimization function, learning, and dropout rate. The proposed ensemble architecture is inspired by Li et al. (2018) [23] and Abdoli et al. (2019) [47], who proposed a deep CNN architecture to learn sound representations directly with changes in the network structure and hyperparameters. Batch Normalization was applied to minimize overfitting, and the dropout method is used after each convolution layer's activation function for features map dimensionality reduction. Finally, a categorical cross-entropy softmax activation function is used as the loss function, with ten output units representing the labeled classes' number. We also apply both 3-cv, 5-cv and 10-cv cross-validation to all methods to determine our deep architecture performance with different folds. Every time, as the training set ($k - 1$), four subsets are combined, and the remaining subset is used for testing. This method is repeated until every subset is used. Table 3 presents the networks structure with initial hyperparameters.

4.5. Evaluation Metrics

Our proposed architecture generalization performance and efficiency require a practical and feasible experimental estimation method and standard measurement. Four

measurement indices, namely: sensitivity, specificity, accuracy, precision, recall, F-measure, area under ROC curve, and the area under the precision-recall curve, were used to measure models performance. Each task output was assessed using different metrics depending on the task. Various performance metrics often contribute to distinct decision outcomes, especially when evaluating various classifiers' capabilities.

$$\text{Sensitivity} = \frac{TP}{TP + FN} \quad (2)$$

$$\text{Specificity} = \frac{TN}{TN + FP} \quad (3)$$

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (4)$$

$$\text{Precision} = \frac{TP}{TP + FP} \quad (5)$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad (6)$$

$$\text{F-measure} = \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (7)$$

The sample is divided into four cases according to the combination of precise classes and algorithm prediction classes: true-positive (TP), false-positive (FP), true-negative (TN), and false-negative (FN). Sensitivity is the probability that groups are correctly categorized. However, specificity is the probability that non-classes are classified correctly. In contrast, precision is the possibility of non-classes being correctly identified. The F-score or F-measure is commonly used to evaluate information in the network. Furthermore, used to combine the precision and recall of the model. Moreover, it measures the networks' accuracy on the selected dataset and evaluates binary classification systems, which classify examples into 'positive' or 'negative'. ROC curves plot y-axis sensitivity and x-axis specificity, corresponding to the decision thresholds. AUROC is indicating the area under the ROC curve, which is the model goodness-of-fit. A perfect model has an AUROC of 1, and random output is indicated by an AUROC of 0.5. The closer AUROC to 1, the better the performance of the model [94].

5. Results and Discussion

5.1. Hardware Requirements

To process and train such a deep model with such a large dataset requires a particular computational effort. This experimental research is being performed on an Intel(R) Core™ desktop workstation with a Core i7-9700 CPU @ 3.40 GHz, a RAM of 32 GB, and the hard drive includes 512 GB SSD. Nvidia GeForce with a GTX 960M model and 8 GB VRAM specifications are the graphics card. The use of the GPU was not needed because of the low computational complexity of the 1D CNN architecture.

5.2. Software Requirements

All of the experiments were performed on Windows 10, Anaconda package management, and python 3.7.9. Different libraries/packages are used in our implementation, such as Pandas, Keras [95]. Proposed models were developed using TensorFlow/Keras [76] to build a baseline of all learning models alongside Scikit-learn [96] for preprocessing data phase. Keras makes it very easy to add, drop, and control layers in our proposed architecture. Librosa [79] was used for audio analysis and features extraction.

5.3. Results

This section explains our experiment results in detail and presents our proposed model efficiency compared to other benchmark approaches and how selected hyperparameters influence our recognition models. The results show that the Bayesian optimization algo-

rithm based on the Gaussian method can achieve higher accuracy in a few data samples based on the experiments we performed compared with manual and other search methods. The performance was assessed over 400 iterations, the initial batch size of 42 samples with early termination, and 12 patients, which is the number of iterations showing no improvement. Early stopping mode was set to “auto” to increase or decrease in validation accuracy automatically.

Based on Bayesian optimization, we selected the top five models over 25 epochs. Several convolutional layers and their number of filters plays an important role in detecting high-level features. It is also noticed that increasing the number of convolution layers will significantly impact the accuracy of the model recognition with waveform input and the four convolution layer networks achieve the best performance. We noticed that increasing the number of stacked network layers under the same features map does not improve recognition accuracy. Convolutional filters search array are set to $\{16, \dots, 512\}$ with a step size of 2. The network uses the strategy of repeated stacking. We used multiple optimizers to optimize the convex and non-convex optimization network, such as Adam, AdaBound, and EAG. We trained the networks with various initial learning rate values (0.1, 0.01, and 0.001), and we found that (0.001) performed better in all models. In order to find the optimum batch size, we compared the models with various batch sizes within the range of $[16, \dots, 256]$, and the batch size of 54 obtained the best results in conducted experiments. Finally, our models adopt 50 training epochs due to no further validation accuracy improvement.

Table 4 summarizes the best configurations for the different models. The epoch's hyperparameters with the highest validation accuracy are determined as the final hyperparameters for each selected model. Bayesian optimization achieved good results after a few iterations, and no significant improvements were apparent in extra iterations. 1D CNN accuracy improved by 2% compare to default configurations.

Table 4. Models selected based on Bayesian optimization.

Model	Filters	Kernels	Batch Size	Activation f	Optimizer	Learning Rate	Dropout Rate
1	141	4	42	ReLU	AdaBound	0.007	0.266
2	100	2	36	—	Adam	0.003	0.181
3	98	2	44	—	EAG	0.005	0.258
4	64	3	58	—	—	0.003	0.110
5	158	3	64	—	Adam	0.002	0.266

Performance on test set by the five models is summarized in Table 5 and the curves for validation set are displayed in Figures 2 and 3. As shown in training curves, the training was finished after 63 epochs due to no further validation accuracy improvement. Model loss decreased over time for both training and validation datasets, indicating that the model learned from the dataset as expected. Similarly, as learning efficiency improved, model training and validation accuracy increased over time. Model loss and accuracy variations over various epochs are due to hidden dropout layers. The gap between curves for training and validation implies that overfitting was minimal; dropout was used to prevent overfitting and improve computational efficiency. The maximum and minimum model loss and accuracy using the validation dataset were (0.353, 0.469) and (88.8%, 91.4%), respectively. The average accuracy of 25 iterations achieved by the 1D CNN on test set in selected models were 89.6%, 91.4%, 90.1%, 89.7%, and 90.0%, respectively, as shown in Table 6.

Figure 2. Epochs vs. loss graphs.

Figure 3. Epochs vs. accuracy graphs.

Table 5. Performance of selected models based on Bayesian optimization.

Model	Loss	Accuracy	Sensitivity	Specificity	Precision	Recall	F-Measure
1	0.738	0.894	0.892	0.989	0.9	0.892	0.896
2	0.751	0.883	0.88	0.987	0.886	0.88	0.883
3	0.689	0.894	0.89	0.989	0.898	0.89	0.894
4	0.808	0.887	0.886	0.988	0.891	0.886	0.889
5	0.654	0.894	0.891	0.989	0.899	0.891	0.895

Table 6. Detailed classification performance based on Bayesian optimization and ensemble learning.

Model	Loss	Accuracy	Sensitivity	Specificity	Precision	Recall	F1
1	0.738	0.894	0.892	0.989	0.9	0.892	0.896
2	0.32	0.915	0.901	0.993	0.93	0.901	0.915
3	0.255	0.924	0.911	0.993	0.939	0.911	0.925
4	0.225	0.934	0.922	0.994	0.946	0.922	0.934
5	0.197	0.944	0.934	0.995	0.954	0.934	0.944

Table 6 and Figure 3 shows the accuracy and loss values of all the above mentioned models under the given number of training epochs. Figure 3, we concluded that the ensemble model's convergence speed is way faster than other method and its trends to convergence with fewer rounds and a lower loss value. These conclusions also demonstrate that the proposed method can guarantee a faster convergence speed, a lower loss value than other methods. Based on the stated evaluation criteria, the output of the proposed 1D CNN on the test results shown in Table 6.

The accuracy of the top five selected classifiers based on Bayesian optimization are 89%, 89%, 92%, 0.93%, 0.94%, respectively. The highest sensitivity of 93.10% and specificity of 99.49% were obtained. The specificity was more than 99% for all validation data. Overall average accuracy obtained was 94.46%, and the maximum accuracy of 95.21% for all experiments iterations. Performance measurements are presented for the weights for which the best results obtained in the study. Figure 4 shows the graphical representation of the comparison of performances for different ensembles.

Figure 4. Comparison of Bayesian and ensemble model.

An ensemble model is developed by integrating five base predictors. Experimental results show that the performance of CNNs is much higher than any individual methods, as shown in Table 6 and Figure 3. Reasons behind ensemble method superiority are: (1) as the essential predictors, methods that produce satisfying results are implemented, and the variety of base predictors helps to construct models of high accuracy ensemble; (2) every base predictor is treated equally by the ensemble model, and no prior information is required.

Moreover, we used a box-plot in Figure 5 to show the accuracy of the different network structures on different subsets under five-fold cross-validations. The results show that different stacked convolutional neural networks' architecture influences their recognition

accuracy. The best recognition accuracy (five-layer network) is 3.2% higher than other 1D CNN approaches.

Figure 5. Box-plot showing Bayesian single model accuracy vs. accuracy of ensemble models.

We further compared the proposed model's performance and explored how the proposed method's information fusion created the confusion matrix and AUC score. From AUC scores, it is observed that the proposed methods produced better results than other methods with scores of 0.99. Figure 6 shows the ROC curves and the calculated AUC scores. While Figure 7a shows the confusion matrix of the Bayesian optimization results, and Figure 7b of ensemble model. Values along the diagonal represent the number of correctly categorized samples for each particular class. Each confusion diagonal matrix displays the prediction accuracy of each sound class out of 10 classes. The comparison demonstrates that the ensemble model recognition accuracy is higher, with an average increase of 4.7%. This implies that the ensemble model can learn higher-level and more features from the raw waveform to improve model prediction performance. On average, the Bayesian optimization and ensemble model has enhanced overall performance by 1% and 5%, respectively.

Confusion matrices indicate that the CH and ST labels are the most challenging classes for 1D CNN. However, EN and GU classes are well classified by the proposed method. DB, CP, and CH are the classes that show relatively low performance. As can be seen in the confusion matrix before and after enabling, the recognition accuracy varies for each sound form. We only show the confusion matrix of the highest Bayesian model and the proposed ensemble model.

Our accuracy reached a maximum of 95.1%, which is higher than that achieved by Abdoli et al. (2019) [47], and Li et al. (2018) [23] with accuracy of 89% and 90.2%, respectively. Furthermore, higher than the 73.7% accuracy of Piczak (2015) [45], and the 79% of Salamon and Bello (2017) [32]. These results indicate that our ensemble 1D CNN has achieved significant sound recognition improvements. It also means that our model structure for ESC with raw waveform input has a higher recognition accuracy. When we applied ensemble theory to merge Bayesian models, the accuracy has increased by an average of 1.26% with each model up to four models and higher accuracy of 5% with the proposed model, proving our model's superiority with the environmental sound events recognition. Our algorithm performance is slightly lower than results reported by Su et al. (2019) [97] of two-stream CNN with the accuracy of 97%. However, that model was 15.9 M number of trainable parameters six times higher than our proposed method. The results of our experiments on the Urbansound8k dataset also confirm our findings. The average

accuracy of the 5-cv on the models before the combination was 90.36% and 94.46% after model ensembling. It also shows that ensemble methods can combine various types of predictions to yield high-accuracy results that outperform current state-of-the-art methods.

Figure 6. Area under the ROC curve (AUC) of predictor model.

Figure 7. Proposed end-to-end Bayesian vs. ensemble model confusion matrix.

To the best of our knowledge, for the first time, the recognition accuracy has surpassed 94.4% using the 1D CNN as a classifier on UrbanSound8K dataset. It proves our approach superiority in environmental sound recognition applications. Compared to other algorithms, experimental results show that the ensemble learning method enhanced the classification accuracy of the 1D CNN by nearly 5%. Compared with other traditional machine learning algorithms, a more robust lifting algorithm, and a deep learning algorithm, the stacking ensemble learning model also has different degrees of improvement. The comparison results for all approaches are summarized in Table 7.

Our main contribution of this paper is a special 1D CNN architecture and its hyperparameter settings using Bayesian optimization and ensemble learning. We trained several different 1D CNNs with various network configurations to analyze different structures impact on recognition performance. First, we tested the 1D CNN architectures with several convolution layers to analyze their influence on the feature extraction. Ensemble multiple models with different configurations are our second main contribution of this work. Because short audio segments do not contain enough information to train the 1D CNNs properly, this study may be specific to the UrbanSound8K dataset and cannot be gener-

alized to other audio classification tasks or datasets. However, it is ideal for mobile or handheld devices with limited power due to the low computational complexity of the 1D CNN architecture.

Table 7. Mean accuracy of different approaches on the UrbanSound8k dataset.

Approach	Year	Representation	Mean Accuracy	# of Parameters
M18 CNN [47]	2017	1D	72%	3.7 M
EnvNet-v2 [63]	2017	1D	78%	101 M
RawNet [23]	2018	1D	87%	377 K
1D CNN Rand [47]	2019	1D	87%	256 K
1D CNN Gamma [47]	2019	1D	89%	550 K
Proposed Ensemble 1D CNN	2021	1D	94%	1.9 M

Future work will explore an adaptive way to automatically adjust the searching process and make the neural network architecture evolve by automatically reshaping, adding, and removing various layers. Furthermore, considering hybrid networks like CNN-LSTM [98] to merge both advantages from each approach has shown much better performance in other domains. Obtaining a network training dataset is the key downside of the neural network. If training data is insufficient and not appropriate, the network will easily learn the dataset bias and generally make poor predictions of data with trends which the training data has never seen. Besides, it is typically more difficult to track back if a network does not function as intended. We have used only UrbanSound8k in this paper for network training, which may lead to bias by overlooking the effects of unusual audio events. Perhaps a more practical approach would be to train the network in a semi-supervised to boost network performance on unseen data by using limited actual data and data generated during the training.

6. Conclusions

This paper proposed an end-to-end Bayesian ensemble one-dimensional CNN for environmental sound classification, archiving higher accuracy with fewer network trainable parameters. It learns the representation from audio waveform directly without additional features extraction and signal processing. We selected the highest performed models based on Bayesian optimization and fused them through the ensemble mechanism. Base-line models are constructed of a convolution layer followed by batch normalization and max-pooling layer with a fully-connected layer and categorical softmax function. Urban-Sound8K benchmarking dataset of 8732 audio samples was used to evaluate the proposed model's performance. Our results indicate that the proposed end-to-end Bayesian Ensemble 1D CNN is superior and more efficient for environment sounds classification applications due to the appropriate hyperparameter selection compared to other state-of-the-art approaches. Sensitivity, specificity, accuracy, precision, recall, F-measure, area under the ROC curve, and the area under the precision-recall curve were used as measurement indices to measure the model's performance. Statistical analysis reveals that the enhancements are significant with a classification accuracy of 94.46% on the UrbanSound8K dataset, which is higher than state-of-the-art end-to-end methods by 5.4%. Future research will look at a more precise method of selecting hyperparameters and additional datasets and repositories. Finding an adaptive way to adjust the searching pounds automatically would be recommended. Furthermore, considering hybrid networks have shown much better performance in other domains.

Author Contributions: Conceptualization, M.G.R.; methodology, M.G.R., S.J.A.; software, M.G.R. and S.J.A.; validation, M.G.R., S.J.A.; formal analysis, M.G.R., N.A., and H.A.; investigation, M.G.R., S.J.A., and A.A.; resources, M.G.R., N.A., H.A., and A.A.; data curation, M.G.R. and A.A.; writing—original draft preparation, M.G.R. and A.B.; writing—review and editing, S.J.A., N.A., H.A., A.B.,

and M.G.R.; visualization, H.A., N.A.; supervision, S.J.A.; project administration, S.J.A.; funding acquisition, S.J.A. All authors have read and agreed to the published version of the manuscript.

Funding: Research reported in this publication was supported by Fundamental Research Grant Project (FRGS) from the Ministry of Education Malaysia (FRGS/1/2018/ICT02/UTP/03/1) under UTP grant number 015MA0-013.

Acknowledgments: We are grateful to the Editor and three anonymous reviewers for their valuable suggestions and comments, which significantly improved the quality of the manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Chan, T.; Chin, C.S. A Comprehensive Review of Polyphonic Sound Event Detection. *IEEE Access* **2020**, *8*, 103339–103373. [\[CrossRef\]](#)
2. Hossain, M.S.; Muhammad, G. Emotion recognition using deep learning approach from audio–visual emotional big data. *Inf. Fusion* **2019**, *49*, 69–78. [\[CrossRef\]](#)
3. Ali, H.; Tran, S.N.; Benetos, E.; Garcez, A.S.D. Speaker recognition with hybrid features from a deep belief network. *Neural Comput. Appl.* **2018**, *29*, 13–19. [\[CrossRef\]](#)
4. Calvo-Zaragoza, J.; Toselli, A.H.; Vidal, E. Handwritten Music Recognition for Mensural notation with convolutional recurrent neural networks. *Pattern Recognit. Lett.* **2019**, *128*, 115–121. [\[CrossRef\]](#)
5. Mydlarz, C.; Salamon, J.; Bello, J.P. The implementation of low-cost urban acoustic monitoring devices. *Appl. Acoust.* **2017**, *117*, 207–218. [\[CrossRef\]](#)
6. Costa, Y.M.; Oliveira, L.S.; Silla, C.N., Jr. An evaluation of convolutional neural networks for music classification using spectrograms. *Appl. Soft Comput.* **2017**, *52*, 28–38. [\[CrossRef\]](#)
7. Laffitte, P.; Wang, Y.; Sodoyer, D.; Girin, L. Assessing the performances of different neural network architectures for the detection of screams and shouts in public transportation. *Expert Syst. Appl.* **2019**, *117*, 29–41. [\[CrossRef\]](#)
8. Almaadeed, N.; Asim, M.; Al-Maadeed, S.; Bouridane, A.; Beghdadi, A. Automatic detection and classification of audio events for road surveillance applications. *Sensors* **2018**, *18*, 1858. [\[CrossRef\]](#) [\[PubMed\]](#)
9. Nanni, L.; Ghidoni, S.; Brahnam, S. Handcrafted vs. non-handcrafted features for computer vision classification. *Pattern Recognit.* **2017**, *71*, 158–172. [\[CrossRef\]](#)
10. Abdulkadir, S.J.; Alhussian, H.; Nazmi, M.; Elsheikh, A.A. Long Short Term Memory Recurrent Network for Standard and Poor's 500 Index Modelling. *Int. J. Eng. Technol.* **2018**, *7*, 25–29. [\[CrossRef\]](#)
11. Balogun, A.O.; Basri, S.; Mahamad, S.; Abdulkadir, S.J.; Almomani, M.A.; Adeyemo, V.E.; Al-Tashi, Q.; Mojeed, H.A.; Imam, A.A.; Bajeh, A.O. Impact of feature selection methods on the predictive performance of software defect prediction models: An extensive empirical study. *Symmetry* **2020**, *12*, 1147. [\[CrossRef\]](#)
12. Abdulkadir, S.J.; Shamsuddin, S.M.; Sallehuddin, R. Moisture prediction in maize using three term back propagation neural network. *Int. J. Environ. Sci. Dev.* **2012**, *3*, 199. [\[CrossRef\]](#)
13. Abdulkadir, S.J.; Yong, S.P.; Foong, O.M. Variants of Particle Swarm Optimization in Enhancing Artificial Neural Networks. *Aust. J. Basic Appl. Sci.* **2013**, *7*, 388–400.
14. Abdulkadir, S.J.; Yong, S.P.; Zakaria, N. Hybrid neural network model for metocean data analysis. *J. Inform. Math. Sci.* **2016**, *8*, 245–251.
15. Abdulkadir, S.J.; Shamsuddin, S.M.; Sallehuddin, R. Three term back propagation network for moisture prediction. In Proceedings of the International Conference on Clean and Green Energy, Dalian, China, 28–30 May 2012; pp. 103–107.
16. Abdulkadir, S.J.; Yong, S.P. Empirical analysis of parallel-NARX recurrent network for long-term chaotic financial forecasting. In Proceedings of the 2014 International Conference on Computer and Information Sciences (ICCOINS), Kuala Lumpur, Malaysia, 3–5 June 2014; pp. 1–6.
17. Abdulkadir, S.J.; Yong, S.P.; Marimuthu, M.; Lai, F.W. Hybridization of ensemble Kalman filter and non-linear auto-regressive neural network for financial forecasting. In *Mining Intelligence and Knowledge Exploration*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 72–81.
18. Abdulkadir, S.J.; Yong, S.P. Scaled UKF–NARX hybrid model for multi-step-ahead forecasting of chaotic time series data. *Soft Comput.* **2015**, *19*, 3479–3496. [\[CrossRef\]](#)
19. Abdulkadir, S.J.; Alhussian, H.; Alzahrani, A.I. Analysis of recurrent neural networks for henon simulated time-series forecasting. *J. Telecommun. Electron. Comput. Eng. (JTEC)* **2018**, *10*, 155–159.
20. Abdulkadir, S.J.; Yong, S.P. Lorenz time-series analysis using a scaled hybrid model. In Proceedings of the 2015 International Symposium on Mathematical Sciences and Computing Research (iSMSC), Ipoh, Malaysia, 19–20 May 2015; pp. 373–378.
21. Abdulkadir, S.J.; Yong, S.P.; Alhussian, H. An enhanced ELMAN–NARX hybrid model for FTSE Bursa Malaysia KLCI index forecasting. In Proceedings of the 2016 3rd International Conference on Computer and Information Sciences (ICCOINS), Kuala Lumpur, Malaysia, 15–17 August 2016; pp. 304–309.

22. Pysal, D.; Abdulkadir, S.J.; Shukri, S.R.M.; Alhussian, H. Classification of children's drawing strategies on touch-screen of seriation objects using a novel deep learning hybrid model. *Alex. Eng. J.* **2021**, *60*, 115–129. [\[CrossRef\]](#)
23. Li, S.; Yao, Y.; Hu, J.; Liu, G.; Yao, X.; Hu, J. An ensemble stacked convolutional neural network model for environmental event sound recognition. *Appl. Sci.* **2018**, *8*, 1152. [\[CrossRef\]](#)
24. Chowdhury, T.H.; Poudel, K.N.; Hu, Y. Time-Frequency Analysis, Denoising, Compression, Segmentation, and Classification of PCG Signals. *IEEE Access* **2020**, *8*, 160882–160890. [\[CrossRef\]](#)
25. Dong, X.; Yin, B.; Cong, Y.; Du, Z.; Huang, X. Environment sound event classification with a two-stream convolutional neural network. *IEEE Access* **2020**, *8*, 125714–125721. [\[CrossRef\]](#)
26. Dogan, S.; Akbal, E.; Tuncer, T. A novel ternary and signum kernelled linear hexadecimal pattern and hybrid feature selection based environmental sound classification method. *Measurement* **2020**, *166*, 108151. [\[CrossRef\]](#)
27. Barchiesi, D.; Giannoulis, D.; Stowell, D.; Plumbley, M.D. Acoustic scene classification: Classifying environments from the sounds they produce. *IEEE Signal Process. Mag.* **2015**, *32*, 16–34. [\[CrossRef\]](#)
28. Demir, F.; Turkoglu, M.; Aslan, M.; Sengur, A. A new pyramidal concatenated CNN approach for environmental sound classification. *Appl. Acoust.* **2020**, *170*, 107520. [\[CrossRef\]](#)
29. Zhang, Y.; Wang, Y.; Zhou, G.; Jin, J.; Wang, B.; Wang, X.; Cichocki, A. Multi-kernel extreme learning machine for EEG classification in brain-computer interfaces. *Expert Syst. Appl.* **2018**, *96*, 302–310. [\[CrossRef\]](#)
30. Ahmad, S.; Agrawal, S.; Joshi, S.; Taran, S.; Bajaj, V.; Demir, F.; Sengur, A. Environmental sound classification using optimum allocation sampling based empirical mode decomposition. *Phys. A Stat. Mech. Its Appl.* **2020**, *537*, 122613. [\[CrossRef\]](#)
31. LeCun, Y.; Boser, B.; Denker, J.S.; Henderson, D.; Howard, R.E.; Hubbard, W.; Jackel, L.D. Backpropagation applied to handwritten zip code recognition. *Neural Comput.* **1989**, *1*, 541–551. [\[CrossRef\]](#)
32. Salamon, J.; Bello, J.P. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Process. Lett.* **2017**, *24*, 279–283. [\[CrossRef\]](#)
33. Zhao, J.; Mao, X.; Chen, L. Speech emotion recognition using deep 1D & 2D CNN LSTM networks. *Biomed. Signal Process. Control* **2019**, *47*, 312–323.
34. Huang, S.; Tang, J.; Dai, J.; Wang, Y. Signal status recognition based on 1DCNN and its feature extraction mechanism analysis. *Sensors* **2019**, *19*, 2018. [\[CrossRef\]](#) [\[PubMed\]](#)
35. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [\[CrossRef\]](#)
36. Kwon, H.; Yoon, H.; Park, K.W. POSTER: Detecting audio adversarial example through audio modification. In Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, London, UK, 11–15 November 2019; pp. 2521–2523.
37. Shin, H.C.; Roth, H.R.; Gao, M.; Lu, L.; Xu, Z.; Nogues, I.; Yao, J.; Mollura, D.; Summers, R.M. Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning. *IEEE Trans. Med Imaging* **2016**, *35*, 1285–1298. [\[CrossRef\]](#)
38. Kwon, H.; Kim, Y.; Yoon, H.; Choi, D. Random untargeted adversarial example on deep neural network. *Symmetry* **2018**, *10*, 738. [\[CrossRef\]](#)
39. Taherdangkoo, R.; Tatomir, A.; Taherdangkoo, M.; Qiu, P.; Sauter, M. Nonlinear autoregressive neural networks to predict hydraulic fracturing fluid leakage into shallow groundwater. *Water* **2020**, *12*, 841. [\[CrossRef\]](#)
40. Bonet-Solà, D.; Alsina-Pagès, R.M. A Comparative Survey of Feature Extraction and Machine Learning Methods in Diverse Acoustic Environments. *Sensors* **2021**, *21*, 1274. [\[CrossRef\]](#) [\[PubMed\]](#)
41. Tatomir, A.; McDermott, C.; Bensabat, J.; Class, H.; Edlmann, K.; Taherdangkoo, R.; Sauter, M. Conceptual model development using a generic Features, Events, and Processes (FEP) database for assessing the potential impact of hydraulic fracturing on groundwater aquifers. *Adv. Geosci.* **2018**, *45*, 185–192. [\[CrossRef\]](#)
42. Hershey, S.; Chaudhuri, S.; Ellis, D.P.; Gemmeke, J.F.; Jansen, A.; Moore, R.C.; Plakal, M.; Platt, D.; Saurous, R.A.; Seybold, B.; et al. CNN architectures for large-scale audio classification. In Proceedings of the 2017 IEEE international Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA, USA, 5–9 March 2017; pp. 131–135.
43. Choi, K.; Fazekas, G.; Cho, K.; Sandler, M. The effects of noisy labels on deep convolutional neural networks for music tagging. *IEEE Trans. Emerg. Top. Comput. Intell.* **2018**, *2*, 139–149. [\[CrossRef\]](#)
44. Ravanelli, M.; Bengio, Y. Speaker recognition from raw waveform with sincnet. In Proceedings of the 2018 IEEE Spoken Language Technology Workshop (SLT), Athens, Greece, 18–21 December 2018; pp. 1021–1028.
45. Piczak, K.J. Environmental sound classification with convolutional neural networks. In Proceedings of the 2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP), Boston, MA, USA, 17–20 September 2015; pp. 1–6.
46. Pons, J.; Serra, X. Randomly weighted CNNs for (music) audio classification. In Proceedings of the ICASSP 2019—2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019; pp. 336–340.
47. Abdoli, S.; Cardinal, P.; Koerich, A.L. End-to-end environmental sound classification using a 1D convolutional neural network. *Expert Syst. Appl.* **2019**, *136*, 252–263. [\[CrossRef\]](#)
48. Su, Y.; Zhang, K.; Wang, J.; Zhou, D.; Madani, K. Performance analysis of multiple aggregated acoustic features for environment sound classification. *Appl. Acoust.* **2020**, *158*, 107050. [\[CrossRef\]](#)
49. Tokozume, Y.; Harada, T. Learning environmental sounds with end-to-end convolutional neural network. In Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA, USA, 5–9 March 2017; pp. 2721–2725.

50. Hoshen, Y.; Weiss, R.J.; Wilson, K.W. Speech acoustic modeling from raw multichannel waveforms. In Proceedings of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), South Brisbane, Australia, 19–24 April 2015; pp. 4624–4628.
51. Pirhooshayan, M.; Snyder, L.V. Forecasting, hindcasting and feature selection of ocean waves via recurrent and sequence-to-sequence networks. *Ocean Eng.* **2020**, *207*, 107424. [\[CrossRef\]](#)
52. Zhou, Q.; Jiang, H.; Wang, J.; Zhou, J. A hybrid model for PM_{2.5} forecasting based on ensemble empirical mode decomposition and a general regression neural network. *Sci. Total Environ.* **2014**, *496*, 264–274. [\[CrossRef\]](#)
53. Salamon, J.; Jacoby, C.; Bello, J.P. A dataset and taxonomy for urban sound research. In Proceedings of the 22nd ACM International Conference on Multimedia, Orlando, FL, USA, 3–7 November 2014; pp. 1041–1044.
54. Salamon, J.; Bello, J.P. Unsupervised feature learning for urban sound classification. In Proceedings of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), South Brisbane, Australia, 19–24 April 2015; pp. 171–175.
55. Zhang, B.; Quan, C.; Ren, F. Study on CNN in the recognition of emotion in audio and images. In Proceedings of the 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS), Okayama, Japan, 26–29 June 2016; pp. 1–5.
56. Schmidhuber, J. Deep learning in neural networks: An overview. *Neural Netw.* **2015**, *61*, 85–117. [\[CrossRef\]](#)
57. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
58. Boddapati, V.; Petef, A.; Rasmussen, J.; Lundberg, L. Classifying environmental sounds using image recognition networks. *Procedia Comput. Sci.* **2017**, *112*, 2048–2056. [\[CrossRef\]](#)
59. Soon, F.C.; Khaw, H.Y.; Chuah, J.H.; Kanesan, J. Hyper-parameters optimisation of deep CNN architecture for vehicle logo recognition. *IET Intell. Transp. Syst.* **2018**, *12*, 939–946. [\[CrossRef\]](#)
60. Kiranyaz, S.; Avci, O.; Abdeljaber, O.; Ince, T.; Gabbouj, M.; Inman, D.J. 1D convolutional neural networks and applications: A survey. *Mech. Syst. Signal Process.* **2021**, *151*, 107398. [\[CrossRef\]](#)
61. Li, X.; Li, J.; Zhao, C.; Qu, Y.; He, D. Gear pitting fault diagnosis with mixed operating conditions based on adaptive 1D separable convolution with residual connection. *Mech. Syst. Signal Process.* **2020**, *142*, 106740. [\[CrossRef\]](#)
62. Davis, S.; Mermelstein, P. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Trans. Acoust. Speech Signal Process.* **1980**, *28*, 357–366. [\[CrossRef\]](#)
63. Dai, W.; Dai, C.; Qu, S.; Li, J.; Das, S. Very deep convolutional neural networks for raw waveforms. In Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA, USA, 5–9 March 2017; pp. 421–425.
64. Palaz, D.; Collobert, R.; Doss, M.M. Estimating phoneme class conditional probabilities from raw speech signal using convolutional neural networks. *arXiv* **2013**, arXiv:1304.1018.
65. Lee, J.; Park, J.; Kim, K.L.; Nam, J. Sample-level deep convolutional neural networks for music auto-tagging using raw waveforms. *arXiv* **2017**, arXiv:1703.01789.
66. Zhu, Z.; Engel, J.H.; Hannun, A. Learning multiscale features directly from waveforms. *arXiv* **2016**, arXiv:1603.09509.
67. Zeghidour, N.; Usunier, N.; Synnaeve, G.; Collobert, R.; Dupoux, E. End-to-end speech recognition from the raw waveform. *arXiv* **2018**, arXiv:1806.07098.
68. Sagi, O.; Rokach, L. Ensemble learning: A survey. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2018**, *8*, e1249. [\[CrossRef\]](#)
69. Rokach, L. *Ensemble Learning: Pattern Classification Using Ensemble Methods*; World Scientific: Singapore, 2019; Volume 85.
70. Rokach, L. Ensemble-based classifiers. *Artif. Intell. Rev.* **2010**, *33*, 1–39. [\[CrossRef\]](#)
71. Zhang, Y.; Miao, D.; Wang, J.; Zhang, Z. A cost-sensitive three-way combination technique for ensemble learning in sentiment classification. *Int. J. Approx. Reason.* **2019**, *105*, 85–97. [\[CrossRef\]](#)
72. Wang, Z.; Srinivasan, R.S. A review of artificial intelligence based building energy use prediction: Contrasting the capabilities of single and ensemble prediction models. *Renew. Sustain. Energy Rev.* **2017**, *75*, 796–808. [\[CrossRef\]](#)
73. Wong, T.T.; Yang, N.Y. Dependency analysis of accuracy estimates in k-fold cross validation. *IEEE Trans. Knowl. Data Eng.* **2017**, *29*, 2417–2427. [\[CrossRef\]](#)
74. Ling, S.T.; Liu, Q.B. New local generalized shift-splitting preconditioners for saddle point problems. *Appl. Math. Comput.* **2017**, *302*, 58–67. [\[CrossRef\]](#)
75. Cherukuri, A.; Mallada, E.; Low, S.; Cortés, J. The role of convexity in saddle-point dynamics: Lyapunov function and robustness. *IEEE Trans. Autom. Control* **2017**, *63*, 2449–2464. [\[CrossRef\]](#)
76. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. Tensorflow: A system for large-scale machine learning. In Proceedings of the 12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16), Savannah, GA, USA, 2–4 November 2016; pp. 265–283.
77. Aguilar-Ortega, M.; Mohino-Herranz, I.; Utrilla-Manso, M.; García-Gómez, J.; Gil-Pita, R.; Rosa-Zurera, M. Multi-microphone acoustic events detection and classification for indoor monitoring. In Proceedings of the 2019 Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA), Poznan, Poland, 18–20 September 2019; pp. 261–266.
78. Awais, A.; Kun, S.; Yu, Y.; Hayat, S.; Ahmed, A.; Tu, T. Speaker recognition using mel frequency cepstral coefficient and locality sensitive hashing. In Proceedings of the 2018 International Conference on Artificial Intelligence and Big Data (ICAIBD), Chengdu, China, 26–28 May 2018; pp. 271–276.

79. McFee, B.; Raffel, C.; Liang, D.; Ellis, D.P.; McVicar, M.; Battenberg, E.; Nieto, O. librosa: Audio and music signal analysis in python. In Proceedings of the 14th Python in Science Conference, Austin, TX, USA, 6–12 July 2015; Volume 8.
80. Chachada, S.; Kuo, C.C.J. Environmental sound recognition: A survey. *APSIPA Trans. Signal Inf. Process.* **2014**, *3*, e14. [\[CrossRef\]](#)
81. Bergstra, J.; Bengio, Y. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.* **2012**, *13*, 281–305.
82. Bhat, P.C.; Prosper, H.B.; Sekmen, S.; Stewart, C. Optimizing event selection with the random grid search. *Comput. Phys. Commun.* **2018**, *228*, 245–257. [\[CrossRef\]](#)
83. Shuai, Y.; Zheng, Y.; Huang, H. Hybrid Software Obsolescence Evaluation Model Based on PCA-SVM-GridSearchCV. In Proceedings of the 2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS), Beijing, China, 23–25 November 2018; pp. 449–453.
84. Levy, E.; David, O.E.; Netanyahu, N.S. Genetic algorithms and deep learning for automatic painter classification. In Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation, Vancouver, BC, Canada, 12–16 July 2014; pp. 1143–1150.
85. Fornarelli, G.; Giaquinto, A. Adaptive particle swarm optimization for CNN associative memories design. *Neurocomputing* **2009**, *72*, 3851–3862. [\[CrossRef\]](#)
86. Syulistyo, A.R.; Purnomo, D.M.J.; Rachmadi, M.F.; Wibowo, A. Particle swarm optimization (PSO) for training optimization on convolutional neural network (CNN). *J. Ilmu Komput. Dan Inf.* **2016**, *9*, 52–58. [\[CrossRef\]](#)
87. Joy, T.T.; Rana, S.; Gupta, S.; Venkatesh, S. Batch Bayesian optimization using multi-scale search. *Knowl. Based Syst.* **2020**, *187*, 104818. [\[CrossRef\]](#)
88. Kolar, D.; Lisjak, D.; Pajak, M.; Gudlin, M. Intelligent Fault Diagnosis of Rotary Machinery by Convolutional Neural Network with Automatic Hyper-Parameters Tuning Using Bayesian Optimization. *Sensors* **2021**, *21*, 2411. [\[CrossRef\]](#)
89. Huang, C.; Yuan, B.; Li, Y.; Yao, X. Automatic parameter tuning using bayesian optimization method. In Proceedings of the 2019 IEEE Congress on Evolutionary Computation (CEC), Wellington, New Zealand, 10–13 June 2019; pp. 2090–2097.
90. Murugan, P. Hyperparameters optimization in deep convolutional neural network/bayesian approach with gaussian process prior. *arXiv* **2017**, arXiv:1712.07233.
91. Mockus, J. *Bayesian Approach to Global Optimization: Theory and Applications*; Springer Science & Business Media: New York, NY, USA, 2012; Volume 37.
92. Bull, A.D. Convergence Rates of Efficient Global Optimization Algorithms. *J. Mach. Learn. Res.* **2011**, *12*, 2879–2904.
93. Ragab, M.G.; Abdulkadir, S.J.; Aziz, N.; Al-Tashi, Q.; Alyousifi, Y.; Alhussian, H.; Alqushaibi, A. A Novel One-Dimensional CNN with Exponential Adaptive Gradients for Air Pollution Index Prediction. *Sustainability* **2020**, *12*, 10090. [\[CrossRef\]](#)
94. Sun, Z.; Jiang, B.; Li, X.; Li, J.; Xiao, K. A Data-Driven Approach for Lithology Identification Based on Parameter-Optimized Ensemble Learning. *Energies* **2020**, *13*, 3903. [\[CrossRef\]](#)
95. Chollet, F. Keras. 2015. Available online: <https://keras.io> (accessed on 1 October 2020).
96. Géron, A. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*; O'Reilly Media: Sebastopol, CA, USA, 2019.
97. Su, Y.; Zhang, K.; Wang, J.; Madani, K. Environment sound classification using a two-stream CNN based on decision-level fusion. *Sensors* **2019**, *19*, 1733. [\[CrossRef\]](#)
98. Li, T.; Hua, M.; Wu, X. A hybrid CNN-LSTM model for forecasting particulate matter (PM_{2.5}). *IEEE Access* **2020**, *8*, 26933–26940. [\[CrossRef\]](#)