Multi-timescale Wavelet Scattering with Genetic Algorithm Feature Selection for Acoustic Scene Classification

Xing Yong Kek¹, Cheng Siong Chin¹, and Ye Li²

¹Faculty of Science, Agriculture & Engineering, Newcastle University, Singapore
²Xylem Water Solutions Singapore Pte Ltd, Decision Science, Singapore

Corresponding author: Xing Yong Kek (e-mail: x.y.kek2@newcastle.ac.uk).

ABSTRACT In this paper, we apply a genetic algorithm (GA) for feature selection, wrapper approach, on wavelet scattering (WS) second-order coefficients to reduce the large frequency dimension (>500). The evaluation demonstrates the capability of GA to reduce the dimension space by approximately 30% while ensuring a minimum performance drop. Furthermore, the reduced WS directly impacts the training time of the convolutional neural network, by reducing the computational time by 20% to 32%. The paper extends its scopes to explore GA for feature selection on multiple timescales of WS: 46ms, 92ms, 185ms, and 371ms. Incorporating multiple timescales has improved classification performance (by ~around 2.5%) as an acoustic representation usually contains information at different time scales. However, it can increase computational cost due to the larger frequency dimension of 1851. With the application of GA for feature selection, the frequency dimension is reduced by 50%, saving around 40% computational time, thus increasing the classification performance by 3% compared to a vanilla WS. Lastly, the entire implementations are evaluated using the Detection and Classification of Acoustic Scenes and Events (DCASE) 2020 dataset, and the proposed multiple timescales model achieves 73.32% of classification accuracy.

INDEX TERMS Acoustic Scene Classification, Convolutional Neural Network, Feature Selection, Genetic Algorithm, Wavelet Scattering

I. INTRODUCTION

Wavelet Scattering (WS) or also known as Deep Scattering Spectrum (DSS), is a signal processing technique developed by Stephane Mallat [1-4] to recover the loss of information present in Log mel-spectrogram, especially when given a timescale larger than 25ms. While research works presented in [5,6] have shown a competitive performance advantage over log mel-spectrogram, the large frequency dimensionality might outweigh the advantage when the computational resource is limited. Hence, there is a need to reduce the frequency dimension of WS without impeding the performance. The task of finding an optimal feature subset can be described as feature selection. In a broader aspect, feature selection is a dimensionality reduction technique. Dimensionality reduction can be categorized into two methods: feature selection and feature construction (extraction). The difference is that the latter usually compares the features in another feature space, and the final output is a new set of features (e.g., Principal Component Analysis (PCA)). In contrast, feature selection selects a subset of the original features.

This paper chooses the feature selection technique as the approach to dimensionality reduction. The reason can be broken down into three factors: Firstly, WS is locally translation invariant and stable to deformation [1-3]. These properties are paramount to acoustic classification tasks. Hence, it is vital to retain the feature as what it is. Secondly, WS consists of first and second-order coefficients, and the first and second-order coefficients have relatively huge differences in magnitude [5,6]. Thus, the translation algorithm might bias the first-order coefficients as their magnitude is larger. Thirdly, the second-order is a sparse representation [1-3], leading to careful selection of the translation algorithm.

Feature selection is broadly studied and has been used in many domains [7-12] with a traditional approach such as exhaustive search, which is practical for a small number of features but not feasible with a large number of features. Exhaustive search requires \(O(2^m)\) computational costs, where \(m\) is the number of features. Hence, employing a heuristic technique [13] is more effective to find the optimal feature subset without searching through every single
combination or NP-Hard problems. Common Heuristic search techniques such as sequential forward/backward selection, "plus-l-take away-r" [14-16], sequential forward/backward floating selection are fast and easy to implement. However, they can only provide local decisions without a global optimal solution. Hence, to expand the search space into a global decision, this paper used a genetic algorithm (GA) [12,14-27], which is a type of evolutionary algorithm (EA), and it emulates biological evolution to find the optimal feature subset. The application of GA for feature selection has been well studied [12,14-27]. Hence, it is appropriate to apply on WS, which has not been used before based on our best knowledge, especially on multiple timescales WS.

Extending to our search for optimal feature subset for WS, this paper proposed combining multiple timescales of WS to further exploit GA’s searching capabilities. The reason for combining multiple timescales of WS is to further reduce the impact of the Heisenberg Uncertainty Principle [6], where one does not know the exact timescale of a given signal. This is especially true for acoustic scene classification (ASC), where a single recording can contain multiple sound sources. However, directly combining multiple timescales of WS will result in an even larger frequency dimension. Hence, the need for GA to eliminate redundant and irrelevant features. As we are combining multiple timescales, the possibility of redundancy and irrelevancy is very high.

Thus, this paper proposed the application of GA for feature selection on WS to reduce the large frequency dimension of WS and combine multiple timescales effectively. Our contributions are as follows:

- We evaluated the effectiveness of GA for feature selection for WS.
- We redefine how multiple timescales of WS can be combined using GA for the feature selection framework.
- We then show the effectiveness of combining multiple timescales for WS.
- The proposed GA implementation can be used on other forms of time-frequency representation.
- We adapted the two-stage convolutional neural network proposed by [6] and further extended the network to process multiple timescales of WS. It highlights the scalability of the two-stage convolutional neural network.

This paper is organized as follows. In Section II, III and IV, we described the algorithms, wavelet scattering, genetic algorithm, and convolutional neural network. The collective implementation of the algorithms is then proposed in Section V and evaluated with DCASE 2020 dataset described in Section VI. The result is then discussed in Section VII. Finally, this paper concludes in Section VIII.

![Figure 1: Construction of Wavelet Scattering with a cascade of wavelet transform and modulus operators](image)

**II. WAVELET SCATTERING**

Wavelet Scattering is constructed with a cascade of wavelet transform and modulus operators, as illustrated in Fig. 1, which has the desirable feature representation properties of being locally translation invariant and stability to time-warping deformation [1,2]. In subsection A, wavelet scattering architecture and algorithm are described. Subsection B discussed the limitation of wavelet scattering timescale and, hence, the reason to incorporate multi-timescale wavelet scattering.

**A. WAVELET SCATTERING**

WS was developed by Stephanie Mallet and presented in [1] as an improved time-frequency representation instead of log mel-spectrogram. WS leverages the cascading of wavelet transform and modulus operators to recover fine-scale information loss through averaging, especially when timescale $T$ is more than 25ms, as observed by [1,2]. The limitation of information loss due to a larger timescale pinpoints the weakness of log mel-spectrogram and the reason for WS. As this paper further exploits WS's multiscale property, the comparison between log mel-spectrogram and WS will not be further elaborated. A comparative analysis of WS and log-mel spectrogram has already been studied in the works of [5,6].

Given a signal $x$, WS can be expressed as cascading convolution of wavelet transform and modulus operations:

$$S_m x(t, \lambda_1, ..., \lambda_m) = |x * \psi_{\lambda_1} * ... * \psi_{\lambda_m} | * \phi(t) \quad (1)$$

where $S_m x$ denotes the output of WS, $m$ is the number of wavelet scattering operations performed on $x$, as illustrated in Fig. 1. The symbol $m$ is usually termed as ‘order’. $t$ is a time index. $\psi$ is the set of wavelet filters with the mother wavelet as 'Morlet' wavelet, and their center frequency is denoted as $\lambda_m$. The number of wavelets per octave of $\psi_{\lambda_m}$ is determined by $Q_m$. Lastly, $\phi(t)$ is an averaging function that acts as a lowpass filter of size $T$. In this context, the Gaussian filter is the lowpass filter. As most of the papers suggested that only the first order and second-order is sufficient to capture the majority of the
signal energy [1-4], this paper will only take the first and second-order, deriving from (1) respectively,

\[ S_1 x(t, \lambda_1) = \{ x \star \psi_{\lambda_1} \} \psi(t) \]  
\[ S_2 x(t, \lambda_1, \lambda_2) = \{ |x \star \psi_{\lambda_1}| \star \psi_{\lambda_2} \} \psi(t) \]

For simplicity, \( S_1 x \) derived in (2) is the first-order coefficients and \( S_2 x \) derived in (3) is the second-order coefficients. \( Q_1 \) and \( Q_2 \) are the number of wavelets per octave for the first and second-orders, respectively. The rest of the paper uses a setup of \( Q_1 = 9 \) and \( Q_2 = 1 \) based on the suggestion of [1-6].

B. GREATER MULTISCALE/MULTIRESOLUTION

Although WS analyzed modules with time-varying filters based on multiple scaled 'Morlet' wavelet as shown in Fig.2, the averaging function that is required for the representation to be invariant to translation will remove fine-scale information depending on the timescale \( T \), in \( S_1 x \), which is then captured by higher orders, mainly \( S_2 x \). In this paper, we look at combining multiple timescales, and \( T \) has a value of \{46,92,185,371\} ms. The selected timescales are based on the observation of [6], where they exhaustively evaluated the classification performance of timescales range between a value of \{23,46,92,185,371,743,1480\} ms for DCASE 2020 Task1a dataset. Their experimental result shows that the effective timescale range from 46ms to 371ms.

It is observed that the larger the \( T \), the more high-frequency information will be removed from the first order after applying the Gaussian filter, as illustrated in Fig.3. The loss information is then captured by using another wavelet transformation. In other words, the larger the \( T \), the more information is captured by the second-order. Hence, supporting the analogy of energy dispersion from first order to second-order at different \( T \)[1-4].

The reason why the high frequency information is retained in the second-order or higher orders, is because the subsequence wavelet modulus transform is applied to \{\( x \star \psi_{\lambda_1} \)\} to achieve sparse representation [1-4]. A sparse representation is a representation with condensed wavelet coefficients with highly concentrated information. Hence, applying an averaging function on higher orders will not lead to further loss of information. The timescale, \( T \) affects the averaging operation, it also affects the number of center frequencies, \( m \), and the bandwidth or scale, of each wavelet filter where they are used to capture the acoustic signal profile.

In summary, the Heisenberg uncertainty principle has some impacts on WS. There is a need to carefully combine multiple timescales to avoid losing fine-scale acoustic signal while preventing redundancy, as shown in Fig 4.

III. GENETIC ALGORITHM

A genetic algorithm is an adaptive heuristic search and optimization algorithm designed with an influence on Darwinism or Darwinian theory [28-31] and is the most common form of EC. GA finds the best solution by a "survival of the fittest" concept, very much like how nature evolves by mixing and selecting the best individuals to reproduce the next generation. The major draw to implementing GA is its mixing and selecting the best individuals to reproduce. The major draw to implementing GA is its mixing and selecting the best individuals to reproduce. The major draw to implementing GA is its mixing and selecting the best individuals to reproduce. The major draw to implementing GA is its mixing and selecting the best individuals to reproduce.

1) INITIALIZATION OF POPULATION

The Population is relatively the pool of selected different combinations of features. Each set of features represents an input representation, and each feature represents an attribute.
of the input representation. Relating to GA and biology, a feature is termed a gene, and a set of features is termed as a chromosome.

The initial step involves the creation of a pool of chromosomes. The chromosomes are created through a random process of combining various genes. Thus, a population is formed containing variants of the chromosome. The Population will then undergo evolution, which is the continuous process of weeding out the weaker chromosomes while encouraging the reproduction between the fittest chromosomes, with the goal of achieving the chromosome with the 'best' genes. As such, the Population will evolve continuously from the initial Population until it meets an end condition.

2) FITNESS FUNCTION
The fitness function is the evaluation algorithm to determine the strength of the chromosomes. Hence, the product of the fitness function is the fitness value.

3) CROSSOVER
Crossover is the technique of mixing the genes of a pair of parents to produce offspring. Both crossover and mutation are techniques used to create new chromosomes based on the selected group of stronger chromosomes from the Population. Hence, the concept of parent and offspring, where the offspring are the newly created chromosomes.

4) MUTATION
Mutation is an alteration in a chromosome. For example, by visualizing a chromosome in a binary format, a chromosome can be represented in a sequence of binaries [0,0,0,1,1,1]. 0 means that the gene in the sequence is not selected, and 1 means that it is selected. Thus, when the mutation occurs, the represented sequence of binaries will flip from 0 to 1, vice versa, from 1 to 0. For example, [0,0,0,1,1,1] can mutate to [1,0,0,0,1,1]. The mutation is crucial to prevent the GA from getting stuck at the local optimal. However, a high mutation rate might impede the GA from finding an optimal solution.

5) END CONDITION
Lastly, the end condition or termination criteria is the algorithm to determine the convergence of GA operation. This can be set as the number of iterations or generations. A generation is the process of step 2 to step 4.

While the application of GA is robust in search optimization [16], we are interested in a particular use case, where GA is used to select the optimal features or for feature selection. The application of GA for feature selection on the wavelet scattering is described in subsection A.

A. GENETIC ALGORITHM FOR FEATURE SELECTION
The concept of using GA for feature selection is not entirely new and has been well established [12,14-27], with a history dating back to 1990 [15,21]. GA for feature selection has the exact workflow as GA described in Section III, but with the objective to maximize or maintain classification accuracy while using a subset of the original features. We have a set of features used in a classification model to perform prediction, and this set of features has large dimensions (e.g., 500 to 1000 features). There is a possibility of redundancy or irrelevancy, which might impede the model from learning correctly and add unnecessary computational complexity.

Hence, GA is employed to search for the optimal features subset based on model performance as the fitness function. Translating the five processes into pseudocode is presented in Table I. GA provides a straightforward mapping to the input representation by encoding each feature with a binary value. 0 means that the feature is not selected and 1 means it is selected.

There are two well-established GA approaches [10]: 'filter approach' and 'wrapper approach', subsequent development is branched into 'embedding' and 'hybrid' approaches [9,15,19]. While the hybrid technique utilized a combination of filter and wrapper approaches [11,12], the embedding approach performs feature selection as part of the pattern recognition algorithm [32].

The difference between the filter and wrapper approach is on the fitness function. The wrapper approach, which is more commonly adopted [15] and evaluates the goodness of the selected features based on the classification model, usually the model's classification accuracy. In GA wrapper approach context, the fitness function relates to the classification algorithm.

| TABLE I
| GA FOR FEATURE SELECTION (WRAPPER) ALGORITHM |
|---------------------------------------------|
| Initialized Population:                     |
| For each h in total_number_of_population do |
|   (a) WS = (F,T) and T' = T.               |
| (b) For each i in F do                     |
|   chromosome[i] = random(0,1)             |
| 2. For each i in total_number_of_generations do |
|   (a) For each j in set_of_chromosomes do |
|   (i) Evaluate the classification accuracy (result is based on cross-validation) |
| (b) Sort Population by classification accuracy |
| Delete i number of chromosomes starting from the bottom. |
| (i = number_of_offspring) (c) Crossover operation: |
| For each k in number_of_offspring do |
|   (i) offspring[k] = parent[k]*1.5 + parent[k+1]*0.5 |
| (d) Mutation operation: |
| For each k in number_of_offspring do |
|   (i) Randomly select genes in offspring[k] based on a given mutation rate. |
| (ii) or each q in selected_genes do |
|   (a) if gene == 1, gene = 0, else gene = 1 |
| (e) Append Population with offspring |

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.315569, IEEE Access
On the other hand, the filter approach uses an independent and less computationally intensive algorithm to evaluate the features subset; only after finding the optimal features subset for the classification algorithm. The objective of the filter approach is to drastically reduce the computational cost of the wrapper approach caused by multiple training and evaluation of the classification model such as deep CNN models. [13,15,16] provide comparative studies between filter approach and wrapper approach, and their findings suggested that wrapper approach outperformed filter approach when comparing classification accuracy. It can be attributed to the core algorithm that evaluates the chromosome’s fitness. The filter approach algorithms such as information theory, correlation-based approach, or distance measure approach might only provide limited interaction between the features [13]. Furthermore, what is deemed insignificant to the filter approach algorithm might be considered useful by the wrapper approach.

Hence, this paper focuses on wrapper approach as we are dealing with WS, which has first and second-order coefficients with very different magnitude and second-order has a sparse representation. Furthermore, a wrapper approach provides the necessary framework to integrate WS with CNN, tapping on the discriminative advantage of CNN [6]. Following the framework presented by [21], we proposed our implementation of GA for WS, which has a meta-heuristic nature to find the global optimal feature subset. Unlike image classification, where features can be flattened into a 1D array and selected pixel-wise, it does not work well for time-frequency representation such as Wavelet scattering. WS has a feature structure [9]. Hence, we proposed the feature selection be performed on the frequency axis. In WS context, we select a subset of wavelet coefficients, and an example is illustrated in Fig. 5. In signal processing, selecting a subset of wavelet coefficients is equivalent to applying bandpass filters on the signal. However, the application of bandpass filters requires strong expert knowledge of the domain, and else one might filter out important information. In the case of ASC, it is even more challenging to decide which frequency spectrums are important as it consists of multiple sound sources and events with various frequency profiles.

This paper proposed two configurations to select the optimal feature subset, and their difference depends on the chosen CNN model. The two CNN models are discussed in Section IV, followed by the two implementations described in Section V.

IV. CONVOLUTION NEURAL NETWORK

As with all wrapper approach, the effectiveness of GA is highly dependable on the classification algorithm. In this paper, the model’s classification accuracy becomes the fitness function. Hence, following the current state-of-the-art framework presented in [33], this paper uses CNN as the classification algorithm and the fitness function. CNN is one of the current leading neural networks that can learn complex structures such as time-frequency representation, images, but not limited to 2-Dimension images [34]. CNN is less computational heavy than the standard neural network or dense network as the nodes or features are not fully connected. Instead, CNN supports weights sharing by having multiple small filters (e.g., 3 x 3) to convolute over a feature map. Hence, each small filter provides a localized understanding of the previous feature map. Coupled with non-linearity activation function and continuous downsampling of the earlier feature maps, CNN can capture low-level abstract features [34], thus learning complex input representation.

The following subsections are delivered as follows, Subsection A presents the two CNN models, and Subsection B elaborates on...
the convolution block design. Lastly, Subsection C discussed other hyper-parameters and configurations applied on the CNN model.

A. TWO-STAGE MULTI-ORDERS CNN
This paper proposed two CNN models for the fitness function of each configuration. The first CNN model is a direct adaptation from [6], and for ease of reference, the Two-stage convolution neural network is abbreviated as (TSCNN). TSCNN is inspired by the work of [35,36], who splits the log mel-spectrogram into frequency bins. The binning process simply divides the log mel-spectrogram into large groups of low, mid, and high-frequency spectrum. Then, in the first stage, the CNN digests this set of frequency bins concurrently in parallel, as illustrated in Fig. 6. In the context of CNN architecture design, this process is termed as ‘group convolution’ [6,34,35,36]. [6] adopted this concept and applied on WS, which conveniently split the first and second-orders. [6] explained that the split is necessary to tackle the large disparity in magnitude between the first and second-order coefficients. Furthermore, they re-terminated group convolution as ‘specialized learning’ as the purpose for group convolution is to allow the network to have independent learning of the first and second-orders before learning them together in the second stage, termed as ‘centralized learning’. The renaming also connects to the second stage where centralized learning is learning the concatenated features of the first and second-orders after understanding them separately. The reason for naming the first stage, specialized learning and the second stage, centralized learning, is mainly for the configuration of the TSCNN. [6] suggested the need to explore various specialized and centralized learning setups, which is reflected in the difference between Fig. 6 and Fig. 7.

TSCNN architecture is suitable for applying mixed first and second-order as the CNN model handles the first and second-order separately. However, the current TSCNN cannot accommodate more timescales, and modification is required. In this paper, we explored the combination of 4 different timescales, hence, \( T = \{46ms, 92ms, 185ms, 371ms\} \). We extended TSCNN to include two additional second-orders [35], making the first stage to be comprised of one first-order and three-second-orders of different timescale. This extended TSCNN is named TSCNN-2, and the architecture is depicted in Fig. 7. Notably, following the suggestion of [6], this paper also performed an ablation study on the number of convolution blocks stacked for specialized learning and centralized learning. Based on the analysis, we have established that additional specialized learning benefited the model and is reflected in Table II and Fig. 7.

B. CONVOLUTION BLOCK DESIGN
Next, this subsection described the convolution block (CB) design which is the fundamental building block of CNN. In other words, CNN is typically built with stacks of similar convolution blocks, except that the number of channels increases as the network goes deeper. Following [6,29] designs, the convolution block is pieced together with the below components as shown in Fig. 6:

1) 3-BY-3 CONVOLUTION LAYER
The 3-by-3 convolution layer (3x3Conv) design has revolutionized most of the current CNN architecture [6,28,29,32,35], which usually follows the combination of stacking 2 3x3Conv. [37] developed 3x3Conv to reduce the computational complexity of the model. They investigated the possibility of reducing the filter size from 7-by-7 or 5-by-5 to 3-by-3 and discovered that a stack of 2 3x3Conv has the same

### Table II

| Component | Description |
|-----------|-------------|
| S1×1      | Specialized Learning |
| S2×2      | Centralized Learning |
| S3×2      |                      |
| S4×2      |                      |

FIGURE 7. TSCNN-2 architecture extended from TSCNN. TSCNN-2 can accommodate 4 WS of different timescales. \( S_1 \times 1, S_2 \times 2, S_3 \times 2 \) & \( S_4 \times 2 \) have timescales of 46ms, 92ms, 185ms and 371ms, respectively. The configuration of specialized and centralized learning differs from TSCNN. Based on our ablation study, TSCNN-2 requires more specialized learning.

FIGURE 8. Convolution Block (CB) design following pre-Normalization residual module approach proposed by [38]. \( x \) denotes the initial feature map and \( x \) denotes the feature map after convolution process. While average pooling is abbreviated as \( \text{AvgP} \) and the product of average pooling is represented as \( \text{AvgP}(x) \).
By propagating the previous gradient forward, degradation no longer happens, even when the network goes extremely deep (e.g., 1000 layers). Similar to 3x3Conv, this residual network architecture is heavily adopted [6, 35, 36].

4) BATCH NORMALIZATION

[39] introduced batch normalization [BN] to accelerate and smoothen the learning process. BN applies a standard normalization on a batch of feature maps using mean and standard deviation. The key difference between a standard normalization and BN is that there are two learning parameters: scaling and offset.

5) ACTIVATION FUNCTION

In this paper, following general selection by most of the authors [6, 34-38], ReLU is a non-linear function, giving:

$$\sigma(x) = \max(0, x) \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases}, \quad (3)$$

is selected as the activation function, which translates all negative values to zero.

6) FULLY COVOLUTIONAL NETWORK

While this is not part of the convolution block design discussed above, 'fully convolution network', better known as FCN [40], is the last convolution block that graciously completes the CNN before the softmax classification layer. It earns its name as FCN as no fully connected or dense layer is used to complete the CNN, commonly found in older models [34, 37, 38]. FCN is constructed with a convolution layer with a filter size of 1-by-1, and the channels are compressed such that it is numerically similar to the number of classes. This was followed by a global average pooling (GAP) that flattened the feature map channel-wise into a vector as the softmax layer.

Lastly, by stacking the convolution blocks together and ending them with FCN, this paper presents the TSCNN-2 architecture in Table II. TSCNN is not shown here as it is a direct adaptation [6].

C. PARAMETERS SETUP

Other than the CNN block and architecture design, other settings are involved when training the model. This subsection is dedicated to providing the information as follow:

1) LEARNING STRATEGY

Following [6, 35, 36] learning strategy, this paper uses stochastic gradient descent (SGD) [41] with warm restart scheduled at a set of epoch indexes being (3, 7, 15, 31, 63, 126, 254) and learning rate covering from 0.1 (max) to 0.00001 (min). Hence, the learning rate will gradually decrease at each step based on half a cosine curve called 'cosine annealing'. This provides adequate learning with a high learning rate to quickly approach local minimal, then a gradual decrease in learning rate to narrow into the local minimum. Based on the indexes derived above, the learning rate is being restarted to ensure the model does not get stuck at the local minimum. [41] further proposed to restart the learning rate with shorter intervals at the beginning of the training phase to accelerate the learning process.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/
FIGURE 9. Process Flow of GATSCNN and GATSCNN-2 implementation. The entire system starts with the initialization of the population described in Fig. 5a and Table II. Next, we evaluate the fitness of all the chromosomes, and depending on the configuration, different CNN models will be selected. Subsequently, we check whether the system has already selected a set of parents. This argument caters to the first iteration, where no parents have chosen yet. Hence, the rest of the iteration will always flow to combining parents and offspring. Then, based on the classification accuracy, we only keep the top-performing chromosomes, and this batch of chromosomes will be the next parents. This will bring us to the next step of creating the offspring based on Table II, step 2. With the offspring, we verify whether the stopping criterion is being satisfied. In this case, our stopping criteria depend on the total number of generations. If we did not meet the criteria, we continue another round of evaluation, selection of new parents, and breeding offspring until we reach our stopping goal. We will have the best optimal feature subset selected by GA upon reaching the stopping goal. Finally, we train and evaluate Mix Up and present the classification result in Section VII.

2) WEIGHT INITIALIZATION AND REGULARIZATION

'He normalization' [42] is being used for weights initialization and subsequently, L2 regularization is being applied.

3) BATCH SIZE AND EPOCH

In this paper, a mini-batch size of 64 and total epochs of 63 are selected for Genetic Algorithm with TSCNN (GATSCNN) and GATSCNN-2 runs. The decision was based on our observation of training TSCNN and TSCNN-2 (without Mix Up) with 254 epochs for at least 10 folds. Their best result usually occurs right at the 62nd Epoch. This phenomenon is due to the effective learning strategy of SGD with warm restart presented in point 1. During the evaluation of the best optimal feature subset, we used a mini-batch size of 32 and 126 epochs as we applied data augmentation, 'Mix Up', described in point 4.

4) DATA AUGMENTATION

Based on analyzing DCASE [33] and suggestions from [6,35,36], this paper adopted the use of 'Mix Up' [43] for data augmentation with an alpha of 0.2 and 0.4 for evaluation of the model alpha for TSCNN-2 and TSCNN, respectively. The value for the alpha is selected based on an exhaustive search where we evaluated alpha ranging from 0.1 to 1 for each model. 'Mix Up' is not applied during GA run as the task of GA is to find the optimal feature subset and might impede the searching process.
The collective combination of WS for feature extraction, GA wrapper approach for feature selection or ‘precise’ bandpass filtering, and CNN as the classifier, is being proposed and depicted in Fig. 10. Figure 9 shows the process flow of the integrated system. For ease of reference, the first configuration is termed ‘GATSCNN’, and the second configuration is ‘GATSCNN-2’.

While GA has the advantage of achieving global optimal [14-21,29-31], it is highlighted that a careful selection of the mutation rate is prudent [19,22]. A mutation operation can be described as changing the binary bit representation from 0 to 1, which means that the selected features based on the parents are flipped. The higher the rate, the more indifferent the offspring are to their parents. Hence, if we apply a high mutation rate for all the offspring, the offspring are more likely to lose important genes from their parents that made them fit in the first place. In other words, we might nullify the effect of crossover, where offspring are created based on the best-performing parents, thus, crippling the heuristic nature of GA. While a low mutation rate might hit local optimal as it becomes highly dependable on the initial Population and the population size, this paper proposed a high mutation rate only for the last two offspring to ensure a global search effect.

Our population initialization strategy is relatively straightforward, and the decision is based on observation from our initial experiments. We initially tested the TSCNN models by randomly picking a subset of features, mimicking the initialization process to hasten the GA process in finding the optimal features. The initial test is equivalent to running the GA once, and the steps are as follow: Firstly, we randomly pick a subset of features based on controlling the number of features to be selected. In this test, we have selected 25%, 50%, 60%, and 70%, sequentially. Based on our GA setup, each selection will consist of 20 different combination of feature subsets; Next, we fed the input representation to TSCNN as depicted in Fig. 6. Our findings brought us to the conclusion of picking 70% of the features as it presents the nearest classification accuracy towards the full features. While for GATSCNN-2, we set the initial number of features selection rate to 50%. This is based on our observation when we tried out various combinations when building ‘FSI’ presented in Section VII, C.

Notably, the number of generations is set at 10 due to the resource limitation of evaluating over a combination of 1000 TSCNN and TSCNN-2 models. In addition, based on a separate experiment, with 20 iterations, the GA no longer demonstrates any gain after the 10th iteration or its ‘plateau’. Hence, in this paper, the GA parameters are presented in TABLE III.

VI. EXPERIMENTAL SETUP

A. DATASET

This paper uses DCASE 2020 challenge task 1a development dataset [33] to evaluate our proposed GA framework. DCASE 2020 task 1a is an ASC challenge where we need to develop a model(s) to predict the whereabouts of an object or person based on an audio recording. The dataset consists of 10-seconds audio segments of 10 acoustic scenes, which are three indoor scenes: airport, shopping mall, metro station, four outdoor scenes: pedestrian street, public square, street traffic, urban park, and three traveling on transport scenes: tram, bus, metro. Each audio data is the mono channel with a 44.1 kHz sampling rate and 24-bit resolution. The development datasets are split into 70% training and 30% testing as per the challenge setup.

B. FEATURE EXTRACTION

The entire 10s raw waveform is then transformed into WS, and following [4-6], we further calculate their deltas and deltas-deltas. Then stacking them together will result in a 3-dimensional input representation of \((F \times T' \times C)\), frequency
TABLE IV
RESULT OF MIXED FIRST AND SECOND-ORDER

| No | \(T_1\) for \(S_1x\) (ms) | \(T_2\) for \(S_2x\) (ms) | ACCURACY |
|----|-----------------|-----------------|----------|
| 1  | 46              | 46              | 65.80%   |
| 2  | 92              | 92              | 70.36%   |
| 3  | 185             | 185             | 70.15%   |
| 4  | 371             | 371             | 69.14%   |
| 5  | 46              | 92              | 70.72%   |
| 6  | 46              | 185             | 70.42%   |
| 7  | 46              | 371             | 70.51%   |
| 8  | 92              | 46              | 69.14%   |
| 9  | 92              | 185             | 70.13%   |
| 10 | 92              | 371             | 69.53%   |
| 11 | 185             | 46              | 68.50%   |
| 12 | 185             | 92              | 70.03%   |
| 13 | 185             | 371             | 69.52%   |
| 14 | 371             | 46              | 68.35%   |
| 15 | 371             | 92              | 68.42%   |
| 16 | 371             | 185             | 69.33%   |

This is the classification result of an exhaustive search of the mixed first and second-order with TSCNN. \(T_1, T_2\) denotes the timescale for the first and second-order, respectively. The first 4 rows are the normal WS, and the rest of the tests are the combination of mixing first and second-order. The highlighted group of mixed first and second-orders have the best classification performance. However, we noticed that the classification result plateau at around ~70%. This finding solidifies our hypothesis that acoustic scene requires even more diversification of timescales, and just mixing two timescales is not enough.

axis, temporal axis, and channel axis comprises the original WS, delta of WS, and deltas-delta.

C. HARDWARE SPECIFICATION

The entire system is implemented using python and run on a Window Desktop with Intel i7-11700F Processor, 32 GB of RAM and RTX 3090.

VII. RESULT

This Section discussed three experimentation results that were evaluated using the DCASE 2020 Task1a dataset described in Section VI. A. Each experiment is trained and tested 5 times, and we calculate the mean of the 5 tests which is presented in the results table. The first experimentation act as a ‘prologue’ to our investigation on the application of GA with WS. In this experimentation, we evaluated all combinations of mixed-WS. The discussion is presented in Subsection A. Based on the findings from Subsection A, we evaluated the possibility of getting an optimal feature subset with GATSCNN in the second experimentation, and it is being discussed in Subsection B. Lastly, taking a step further, we explored the application of GA in achieving optimal feature subset when four timescales are being combined together. The experimentation result is discussed in Subsection C.

A. STUDY OF MIXED FIRST AND SECOND-ORDER

Prior to conducting the experimentation presented in Subsections B and C, we exhaustively searched through a combination of the first and second-order of different timescales as suggested by [44]. In this experiment, we paired a set of first-order with timescales, \(T_1 = \{46, 92, 185, 371\}\)ms with a set of second-order with \(T_2\) having a similar value as \(T_1\), but satisfy the clause \(T_1 > T_2\). The best combination of first and second-orders will then be selected for the GA feature selection. Based on the observation of Table IV, the best combination occurs when a small \(T_1 (46ms)\) is combined with any \(T_2 > 46ms\). While combining the first and second-order the other way around, where \(T_1 > T_2\), in most cases, it will result in poorer performance than not combining them (Table IV, rows 1, 2 & 3). This finding is consistent with the theory of energy dispersion from first to second-order as \(T\) gets larger, suggested by [1-3]. In addition, the combination of timescales with the first order being the smallest \(T\), present better result (Table IV rows, 5, 6 & 7). While the combination of timescales with the first order being the largest \(T\), present poorer result (Table IV rows, 11, 14, 15 & 16). Hence, based on the results, we derived a general rule of \(T_1 < T_2\) when constructing a ‘mixed’ first and second-order. Another observation is that mixing of the first or second-order of timescale of 46ms with larger timescale will result in a considerable boost of approximately ~4% in classification performance, as shown in Table IV, row 1, compared to Table IV, rows 5, 6, 7, 8, 11, 14. This phenomenon can be attributed to the acoustic profile of the acoustic scene, where important acoustic information is mainly captured by a larger timescale, \(T>46ms\). However, the papers [1-3] do not indicate the exact values and the optimal combinations presented in this experiment.

B. GATSCNN EVALUATION

The findings from Subsection B perpetuate the evaluation of GATSCNN using features from Table IV, rows 5, 6 & 7. The experimentation results are tabulated in Table V. Notably, our experiments demonstrated that GA with the first order did not amount to any improvement. As such, GA for the first order is not being evaluated further. Hence, for experiments GATSCNN and GATSCNN-2, GA is only applied on the second-order. The application of GA has shown optimistic
In the next experiment, we evaluate the combination of all the features from Table V, rows 4, 5 & 6. Following Table V, the feature subset only reflects the total number of features for the second-order. The algorithm for the construction of FS1 is described in subsection C. FS2 is the best solution from GATSCNN-2. Lastly, FSALL is the full set of features from the three timescales.

Results for Table V, row 5, while for Table V, rows 4 & 6, there is a slight decrease in accuracy of approximately 0.3%. However, for all cases, Table V, rows 4, 5 & 6, GA has managed to reduce the dimension of the features by at least ~27%, which translates to a reduction in model training cost of 20s-25s per epoch presented in Table VII. Hence, we reaffirmed that the second-order coefficients of WS consist of redundancy and, for the case of Table V, row 5, irrelevancy too.

Redundancy can likely occur in second-order coefficients as it is a more refined representation of first order coefficients (see (3)). Hence, there lies a possibility that a sound source is captured in multiple second-order coefficients. On the other hand, irrelevancy occurs when no important sound sources reside in that frequency spectrum, or a frequency spectrum shares similar second-order coefficients with the rest of the data. Thus, rendering the second-order coefficients not useful when discriminating different acoustic scenes.

Irrelevancy is evident when we combine the models by averaging the final logits from the softmax layer, as shown in Table V, rows 8 & 9. In the comparison of Table V, rows 8 & 9 have shown an improvement of 1.30% classification accuracy, on top of a combined computational cost reduction of 31s to 45s per epoch. While combining just Table V, rows 4 & 5, models have shown the best result for GATSCNN implementation.

TABLE VI

| No | Features | MODEL | FEATURE SUBSET | GA | ACCURACY |
|----|----------|-------|----------------|----|----------|
| 1  | FS1      | TSCNN | 920            | -  | 72.60%   |
| 2  | FS2      | TSCNN | 923            | Yes| 72.89%   |
| 3  | FSALL    | TSCNN | 1851           | No | 72.32%   |
| 4  | FS1      | TSCNN-2| 920         | -  | 71.55%   |
| 5  | FS2      | TSCNN-2| 923          | Yes| 73.32%   |
| 6  | FSALL    | TSCNN-2| 1851        | No | 72.36%   |

This is the classification result of GATSCNN-2 and a comparative study of FS1 vs FS2 vs FSALL. FS1 is constructed by concatenating the feature subsets from Table V, rows 4, 5 & 6. Following Table V, the feature subset only reflects the total number of features for the second-order. The algorithm for the construction of FS1 is described in subsection C. FS2 is the best solution from GATSCNN-2. Lastly, FSALL is the full set of features from the three timescales.

C. GATSCNN-2 EVALUATION

In the next experiment, we evaluate the combination of all the timescales selected in this paper, being 46ms,92ms,185ms and 371ms as mentioned in Section III, B. While evaluating a single second-order coefficient in Subsection B has highlighted the existence of redundancy and irrelevancy, the effect is indeed much greater when we combine all the timescales, as shown in Table VI. Prior to evaluating GATSCNN-2, we construct another feature with all the timescales based on the second-order feature subsets from Table V, rows 4, 5 & 6, and termed it as ‘FS1’. FS1 is constructed using a statistical-based approach on each timescale.

TABLE VII

| No | Features | TIME(s) | FS TIME(s) |
|----|----------|---------|------------|
| 1  | 46-92    | 75      | 53         |
| 2  | 46-185   | 84      | 57         |
| 3  | 46-371   | 98      | 77         |
| 4  | ALL      | 200     | 120        |

This table presents the computational time required before and after feature selection, column TIME, and FS TIME, respectively. The computational time is recorded based on the average time to train each epoch with a batch size of 32.

TABLE VIII

| No | Ref | FEAT | DATA AUG | MODEL | EVAL/DEV ACC |
|----|-----|------|----------|-------|--------------|
| 1  | [45]| LMS  | Mix Up Temporal Crop | Ensemble TSCNN-liked [36] | 76.5%/74.2% |
| 2  | [46]| LMS  | Reverberation + DRC Mix Up SpecAug | Ensemble VGG-liked [34] and TSCNN-liked [36] | 76.2%/81.9% |
| 3  | [47]| LMS  | Mix Up Temporal Crop | Ensemble [36], [36]+focal loss and [36]+auxiliary classifier | 75.0%/71.7% |
| 4  | *   | WS   | Mix Up | GATSCNN-2 | -/73.3%      |

This table presents the top 3 models from DCASE 2020 Task1a. Starting after the index column, ’Ref’ points to the reference of the paper. ’FEAT’ stands for feature, and under feature, LMS stands for log mel-spectrogram. Next column, ’DATA AUG’, stands for Data Augmentation and column ’EVAL/DEV ACC’ stands for classification accuracy for the evaluation dataset and development dataset. Under column ’DATA AUG’, ’SpecAug’ stands for spectrum augmentation, which involves randomly masking of the frequency axis and time axis before training. The DRC stands for Dynamic Range Compression. We denote ‘*Etc’ for [46], as they used several data augmentation techniques. Under column ’MODEL’, we included the result of a single model labelled ‘SINGLE’. This is to provide another perspective when comparing our Genetic Algorithm with Two-stage Convolution Neural Network (GATSCNN-2) model. Notably, most of the models adapted a Two-staged Convolution Neural Network (TSCNN)-like or TSCNN variants approach, and all of them uses log mel-spectrogram. While not stated in the table, all the models use the same learning strategy indicated in Section IV, C, point 1.

Let use Table V, row 4, as an example. The final product of GATSCNN provides the best solution and a set of parents. This set of parents can be said as the ‘best solutions’ from the GA run, and the best solution can be expressed as ‘the best of the best’. With the best solutions or parents, we calculate the number of occurrences each feature or gene is being selected. Hence, in this context, a highly desired feature will have a frequency count of 8 and the worst case will have a frequency
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3156569, IEEE Access

Author Name: Preparation of Papers for IEEE Access (February 2022)
suppressed by 2.9% on the development dataset. The estimated gain can range from 0.7% to 1.2% as there is no classification accuracy recorded for the single model[47]. By estimating using the available ensemble results, there is a possibility of the higher classification accuracy contributed by the multiple data augmentation techniques despite increasing the training time leading to higher computing resources.

Based on our comparative study, the best way to improve the classification accuracy is to enhance the frequency resolution. However, doing so will have a caveat, the increase in frequency resolution will directly translate to an increase in computational cost as shown in Table IX, row 2 compared to row 1, and for WS, row 4 to row 6. While Table IX presented a computational time complexity comparison between the features, it is still not ‘fair’ to directly compare LMS against WS as the CNN architecture plays a significant role in contributing to the computational time complexity. Hence, Table IX is mainly used to compare LMS against higher resolution LMS and WS with GA feature selection against without, separately. In addition, it also provides a referencing towards the computationally time competitiveness of our proposed GATSCNN-2.

In summary, we presented a fresh approach to using GA feature selection to combine multi-timescales WS. It has shown that it improves the classification accuracy performances and makes multi-timescales WS computationally time competitive. This approach was not realized in [45-47]. Their collectively proposed methods include a direct increase in frequency resolution, improved CNN architecture, improved loss function, incorporating, and an ensemble model. All proposed approaches improve the classification accuracy at the price of added computational time.

VIII. CONCLUSION

In this paper, we evaluated the effectiveness of genetic algorithm (GA) for feature selection on wavelet scattering (WS) using a wrapper approach. Our results suggested that the second-order of WS indeed consists of redundant features, and redundancy was amplified when multiple timescales were combined. We reduced the frequency dimension by at least 27% while maintaining the classification accuracy at around 70%. Feature irrelevancy was spotted when we combined the first order with a timescale of 46ms and a second-order with a timescale of 185ms. By removing unnecessary features, we achieved an increase in classification accuracy of 1%, and our best Genetic Algorithm with Two-stage Convolution Neural Network (GATSCNN) model has a classification accuracy of 71.12%. With the successive result from GATSCNN, this paper proposed GATSCNN-2 to combine multiple timescales effectively. GA has reduced the number of features from 1851 to 923, which is equivalent to a 50% reduction. It also improved the classification accuracy by 1%, thus, presenting this paper best model with a classification accuracy of 73.32%.

In addition, the reduction in frequency dimension directly translates to a saving in computational time of 10% and 40%, GATSCNN and GATSCNN-2, respectively. While we only present the time saved for training with CNN, its effects should also decrease the time complexity when constructing WS. However, the disadvantage of all wrapper approaches is the computational cost [7,9-11,13,16,17,21-23]. Hence, there is a need to explore a more computational efficient implementation with GA approach. Yet another study could branch into multi-objective GA where minimization of the feature subset size and logloss should be part of the fitness function.

REFERENCES

[1] S. Mallat, “Group Invariant Scattering,” Communications on Pure and Applied Mathematics, vol. 65, 2012, DOI: 10.1002/cpa.21413.
[2] J. Andén, and S. Mallat, “Deep Scattering Spectrum,” IEEE Transactions on Signal Processing, vol. 62, pp. 4114-4128, 2014.
[3] J. Brun, and S. Mallat, “Invariant Scattering Convolution Networks,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, pp. 1872-1886, 2013.
[4] J. Andén, and S. Mallat, “Multiscale Scattering for Audio Classification,” ISMIR, 2011.
[5] V. Peddinti, T. Sainath, S. Maymon, B. Ramabhadran, D. Nahamoo, and V. Goel, “Deep Scattering Spectrum with deep neural networks,” IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 210-214, 2014.
[6] X. Y. Kek, C. S. Chin and Y. Li, “An Investigation on Multiscale Normalised Deep Scattering Spectrum with Deep Residual Network for Acoustic Scene Classification,” In Proceeding 22nd IEEE/ACIS International Summer Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, 2021.
[7] H. Liu, and Z. Zhao, “Manipulating Data and Dimension Reduction Methods: Feature Selection,” Encyclopedia of Complexity and Systems Science, 2009.
[8] R. V. Jorge and A. Estévez. Pablo, “A review of feature selection methods based on mutual information.” Neural Computing and Applications 24, pp. 175-186, 2013.
[9] Tang, J., Alelyani, S. and Liu, H., “Feature selection for classification: A review,” Data Classification: Algorithms and Applications. CRC Press, pp. 37-64, 2014.
[10] P. Langley, “Selection of Relevant Features in Machine Learning,” 1994.
[11] M. Dash, and H. Liu, “Feature Selection for Classification,” Intelligent Data Analysis 1, pp. 131-156, 1997.
[12] A. K. Uysal, “On Two-Stage Feature Selection Methods for Text Classification,” in IEEE Access, vol. 6, pp. 43323-433251, 2018, doi: 10.1109/ACCESS.2018.2863547.
[13] B. de la Iglesia, “Evolutionary computation for feature selection in classification problems,” WIREs. Data Mining Knowl Discov, 3, pp. 381-407, 2013.
[14] B. Xue, L. Cervante, L. Shang, W. N. Browne, and M. Zhang, “Multi-objective Evolutionary Algorithms for filter Based Feature Selection in Classification,” International Journal on Artificial Intelligence Tools, vol. 22, no. 4, 2013, DOI: 10.1142/S0218081713500243.
[15] B. Xue, M. Zhang, W. N. Browne and X. Yao, “A Survey on Evolutionary Computation Approaches to Feature Selection,” in IEEE Transactions on Evolutionary Computation, vol. 20, no. 4, pp. 606-626, Aug. 2016, DOI: 10.1109/TEVC.2015.2504420.
[16] Y. Saedis, I. Inza, P. Larrañaga, “A review of feature selection techniques in bioinformatics,” Bioinformatics, Vol. 23, pp. 2507-2517, Oct, 2007, doi: 10.1093/bioinformatics/btm344.
[17] S. Oreski and G. Oreski, “Genetic algorithm-based heuristic for feature selection in risk assessment,” Expert Systems with Applications, vol. 41, no. 4, pp. 2052-2064, 2014, DOI: 10.1016/j.eswa.2013.09.004.
[18] A. N. Sloss and S. Gustafson, “2019 Evolutionary Algorithms Review.” Genetic Programming Theory and Practice XVII. Genetic
and Evolutionary Computation. Springer, Cham, 2020, DOI: 10.1007/978-3-030-39958-0_16

[19] J. Yang and Y. Honavar, “Feature Subset Selection Using a Genetic Algorithm,” Feature Extraction, Construction and Selection. The Springer International Series in Engineering and Computer Science, vol. 453, Boston, MA, 1998, DOI: 10.1007/978-1-4615-5725-8_8

[20] M. Jung and J. Ziehe, "A Guided Hybrid Genetic Algorithm for Feature Selection with Expensive Cost Functions," Procedia Computer Science, vol. 18, pp. 2337-2346, 2013, DOI: 10.1016/j.procs.2013.05.405.

[21] W. Siedlecki and J. Sklansky, “A note on genetic algorithms for large-scale feature selection,” Pattern Recognition Letters, vol. 10, no. 5, pp. 335-347, 1989, DOI: 10.1016/0167-8655(89)90037-8.

[22] J.-H. Hong and S.-B. Cho, “Efficient huge-scale feature selection with specified genetic algorithm,” Pattern Recognition Letters, vol. 27, no. 2, pp. 143-150, 2006, DOI: 10.1016/j.patrec.2005.07.009.

[23] T. Zhang, B. Ding, X. Zhao and Q. Yue, “A Fast Feature Selection Algorithm Based on Swarm Intelligence in Acoustic Defect Detection,” in IEEE Access, vol. 6, pp. 28848-28858, 2018, DOI: 10.1109/ACCESS.2018.2833164.

[24] G. J. Ansari, J. H. Shah, M. C. Q. Arazias, M. Sharif, N. Qadeer and H. U. Khan, “An Optimized Feature Selection Technique in Diversified Natural Scene Text for Classification Using Genetic Algorithm,” in IEEE Access, vol. 9, pp. 54923-54937, 2021, DOI: 10.1109/ACCESS.2021.3071169.

[25] E. Alexandre, L. Cuadra, M. Rosa and F. Lopez-Ferreras, “Feature Selection for Sound Classification in Hearing Aids Through Restricted Search Driven by Genetic Algorithms,” in IEEE Transactions on Audio, Speech, and Language Processing, vol. 15, no. 8, pp. 2249-2256, Nov. 2007, DOI: 10.1109/TASL.2007.905139.

[26] A. B. Ibrahim et al., “Optimizing Arabic Speech Distinctive Phonetic Features and Phoneme Recognition Using Genetic Algorithm,” in IEEE Access, vol. 8, pp. 200395-200411, 2020, DOI: 10.1109/ACCESS.2020.3034762.

[27] M. H. Nguyen, P. Le Nguyen, K. Nguyen, V. A. Le, T.-H. Nguyen and Y. Ji, “PM2.5 Prediction Using Genetic Algorithm-Based Feature Selection and Encoder-Decoder Model,” in IEEE Access, vol. 9, pp. 57338-57350, 2021, DOI: 10.1109/ACCESS.2021.3072280.

[28] J. H. Holland, “Genetic Algorithms and Adaptation,” Adaptive Control of Ill-Defined Systems. NATO Conference Series (II Systems Science), vol. 16. Springer, Boston, MA, 1984, DOI: 10.1007/978-1-4684-8941-5_21

[29] D. E. Goldberg, “The Design of Innovation,” Technological Forecasting and Social Change, vol. 64, pp. 7-12, 2000, DOI: 10.1016/S0040-1625(99)00079-7.

[30] D. E. Goldberg. “Genetic Algorithms in Search, Optimization and Machine Learning” (1st ed.). Addison-Wesley Longman Publishing Co., Inc., USA, 1989.

[31] C. Darwin, “On the Origin of the Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life,” London, Nov, 1859.

[32] N. Spolaor, A. C. Lorena, H. D. Lee, “Multi-objective Genetic Algorithm Evaluation in Feature Selection,” Evolutionary Multi-Criterion Optimization. EMO 2011. Lecture Notes in Computer Science, vol. 6576, 2011, DOI: 10.1007/978-3-642-19893-9_32

[33] T. Heittola, A. Mesaros, and T. Virtanen, “Acoustic scene classification in dcase 2020 challenge: generalization across devices and low complexity solutions,” in Proceedings of the Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DACASE2020), pp. 56-60, 2020, URL: https://arxiv.org/abs/2005.14623.

[34] A. Krizhevsky, “One weird trick for parallelizing convolutional neural networks,” ArXiv, 2014, abs/1404.5997.

[35] S. S. Phaye, E. Benetos, Y. Wang, “SubSpectralNet – Using Sub-spectrogram Based Convolutional Neural Networks for Acoustic Scene Classification.” ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 825-829, 2019.

[36] M. D. McDonnell and W. Gao, “Acoustic Scene Classification Using Deep Residual Networks with Late Fusion of Separated High and Low Frequency Paths,” IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, pp. 141-145, 2020, DOI: 10.1109/ICASSP40776.2020.9053274.

[37] K. Simonyan, and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” CoRR, 2015, abs/1409.1556

[38] K. He, X. Zhang, S. Ren and J. Sun, “Identity Mappings in Deep Residual Networks,” ArXiv, 2016, abs/1603.05027.

[39] J. Sergey and C. Song, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift." ICML’15: Proceedings of the 32nd International Conference on International Conference on Machine Learning, vol. 37, pp. 448-456, Jul, 2015.

[40] E. Shelhamer, J. Long, and T. Darrell, “Fully Convolutional Networks for Semantic Segmentation,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 4, pp. 640-651, 2017.

[41] I. Loshchilov and F. Hutter, “SGDR: stochastic gradient descent with restarts,” CoRR, 2016, abs/1608.03983.

[42] K. He, X. Zhang, S. Ren and J. Sun, “Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification,” IEEE International Conference on Computer Vision (ICCV), 2015, DOI: 10.1109/icc.v.2015.123.

[43] H. Zhang, M. Cissé, Y. Dauphin and D. Lopez-Paz, “mixup: Beyond Empirical Risk Minimization,” ArXiv, 2017, abs/1710.09412.

[44] Y. X. Kek, C. S. Chin, Y. Li, “A Low-complexity Computing based on Wavelet Scattering with Mobile Shuffling Network for Acoustic Scene Classification,” unpublished.

[45] S. Suh, S. Park, Y. Jeong, T. Lee, “Designing Acoustic Scene Classification Models with CNN Variants,” DCASE2020 Challenge, Tech. Rep., Jun, 2020.

[46] H. Hu, C. H. Yang, X. Xia, X. Bai, X. Tang, Y. Wang, S. Niu, L. Chai, J. Li, X. Zhao, F. Bao, Y. Zhao, S. M. Siniscalchi, Y. Wang, J. Du, C. Lee, “Device-Robust Acoustic Scene Classification Based on Two-stage Categorization and Data Augmentation,” DCASE 2020 Challenge, Tech. Rep., Jun. 2020.

[47] W. Gao and M. D. McDonnell, “Acoustic Scene Classification using Deep Residual Networks with Focal Loss and Mild Domain Adaptation,” DCASE 2020 Challenge, Tech. Rep., Jun. 2020.

---

**XING YONG KEE** was born in Singapore in 1986. He received the B.S. degrees in computer science from the University of London, Goldsmiths College, Singapore, in 2011 and M.S. degree in knowledge engineering from National University of Singapore, Singapore, in 2017. He is currently pursuing the Ph.D. degree in science, agriculture and engineering at Newcastle University, Singapore.

From 2011 to 2015, He has been a Software Engineer. From 2015 to 2017, He has been a Team/Technical Lead and from 2017 to 2018, He has been a Senior Data Engineer. Since 2018, he has been working at Xylem Inc as a Data Scientist. His research interests include acoustic signal processing, deep learning, machine learning, and evolutionary computing.

**CHENG SIONG CHIN** (M’01–SM’09) became a Member of IEEE since 2001. He received his Ph.D in Applied Control Engineering at Research Robotics Centre in Nanyang Technological University (NTU) in 2008 and M.Sc. (Distinction) in Advanced Control and Systems Engineering from The University of Manchester in 2001. He also graduated with a B.Eng (Hons) in Mechanical and Aerospace Engineering from NTU in 2000.

He is currently an Associate Professor (Reader) at Newcastle University on the Singapore campus. He was a Lecturer in Mechatronics Engineering at Temasek Polytechnic. He worked in the consumer electronics industry few years before moving into academia. He
currently holds 3 U.S. Patents in the area of electronics test systems and components. His research interests include designing and simulating complex systems in an uncertain environment such as marine electric vehicles, energy storage systems, and acoustic systems.

He is an author/co-author of over 100 peer-reviewed papers on journals and conference proceedings. He is an Associate Editor for IEEE Access and IEEE Transportation Electrification Community (TEC) eNewsletter. He has served as General and Publicity Chair in various international conferences.

YE LI received the master’s degree in Mechatronics from University of Southern Denmark in 2009, and the Ph. D. degree from the Biomedical Engineering, Technical University of Denmark in 2012.

He is currently a Senior Data Scientist with Xylem Water Solutions Singapore. He has years of experience in signal processing, image processing, building applications for Internet of Things (IoT). His primary focus is sensor fusion, anomaly detection, time-series analysis.