An Intelligent Low-Complexity Computing Interleaving Wavelet Scattering Based Mobile Shuffling Network for Acoustic Scene Classification

XING YONG KEK¹, CHENG SIONG CHIN¹, (Senior Member, IEEE), AND YE LI²

¹Faculty of Science, Agriculture and Engineering, Newcastle University, Singapore 609607
²Xylem Water Solutions Singapore Pte Ltd., Singapore 609935
Corresponding author: Xing Yong Kek (x.y.kek2@newcastle.ac.uk)

ABSTRACT The key towards a low complexity model for convolution neural network is in controlling the number of parameters of the network and ensuring that the input representation is not extremely large. Hence, to tackle low complexity for acoustic scene classification (ASC), this paper proposed an enhanced wavelet scattering representation with a combination of mobile network modules and shuffling modules. While wavelet scattering comprises wavelet transform with multiple wavelet scales, the averaging operation to make the wavelet scattering invariant to translation limit the maximum timescale. Hence, wavelet scattering is affected by Heisenberg’s Uncertainty Principle. However, creating an input representation with multiple timescales does not meet the brief of low complexity modelling. Hence, we proposed a simple mixing of the first and second order with different timescales. The result is an input representation with nearly the same dimension as the usual wavelet scattering but with enhanced multiscale. To further leverage the ‘interleaved’ wavelet scattering, this paper presents sub-spectral shuffling inspired by shuffling modules that use stochasticity to improve the model’s generalization. Unlike channel shuffle that shuffles channel-wise and spatial shuffle that shuffles pixel-wise, sub-spectral shuffle aims at shuffling the feature maps frequency-wise with the concept of binning. Each bin is shuffled, so the high-frequency spectrum is shuffled to low-frequency spectrum position. As such, the model learns the general acoustic profile of a scene rather than memorizing what is happening at the low-frequency or high-frequency spectrum is erratic for ASC. In addition, this paper also studied temporal shuffling, which shuffles the feature maps temporal-wise, and evaluated sub-spectral shuffling, temporal shuffling, and channel shuffling individually. Our results demonstrated the superiority of sub-spectral shuffling and the modularity of shuffling modules. We then evaluate various combinations of the three shuffling modules on three acoustic scene classification datasets. Our best model combines the three shuffling modules and achieves 70.6% classification accuracy on DCASE 2021 Task 1a dataset, 82.15% on ESC-50 dataset, 81% on Urbansound8K, with ~65K parameters and a size of 126.6KB. In addition, the inclusion of shuffling modules has increased the model performance. Sub-spectral shuffling is especially useful in improving logloss, a metric used to determine the confidence level of the model.

INDEX TERMS Acoustic scene classification, convolutional neural network, mobile network, wavelet scattering.

I. INTRODUCTION

In recent Acoustic Scene Classification (ASC) challenge hosted by the Detection and Classification of Acoustic Scenes and Events community (DCASE) [1], there is a trend toward low-complexity computing. Acoustic Scene Classification (ASC) or Environmental Sound Classification (ESC), is the tasks of recognizing the vicinity you are at, based on the sound produced by the scene. The use case of low computational complexity model is in line with the advancement and
mass adaptation of the Internet of Things and robotics, such as in hearing-aid [2], [3]. On-node detection is paramount to making near real-time decisions. Therefore, low latency and less memory model is desired. Low complexity convolution neural network (CNN) model is not new and a survey on model compression is presented [4]. Current strategies include parameter pruning [5], knowledge distillation [6], quantization [7], and convolution factorization [8].

Parameters pruning [7] deletes nodes deemed least important by the algorithms. The most popular approach is based on the magnitude of the nodes’ weight or the gradients. The magnitude signifies the importance of the node to the model (Large magnitude means more important).

Knowledge distillation [6] tackles low complexity with a “teacher” and “student” network concept. The student network is the desired low complexity model, and the teacher network is a larger network with the desired classification accuracy. In simplicity, the concept involves the teacher guiding the student, and corresponding to the neural network context, the teacher network calculated loss has an impact on the student final loss function and the level of influence is usually determined by a rate between $0$ to $< 1$.

The quantization [7] technique usually reduces computer memory size based on the data type. Commonly, the default data type for training deep networks is 32-bit floating points, which translates to each network parameter having memory upkeep of 4 bytes. Hence, converting to a smaller memory data type (e.g., 16-bit floating points, 8-bit integer) will drastically reduce the overall memory size of deep networks.

Next, convolution factorization is the concept of adapting matrix factorization to the convolution layer filters [8]. Such design has been around in the image domain with state-of-the-art models, such as InceptionNet [9], MobileNet [10], [11] and ShuffleNet [12]. InceptionNet suggested that the convolution operations are a matrix multiplication between two tensors and have the same properties as the matrix. Hence, it can be decomposed/factorized. The proposed ‘filter-wise’ factorization, which breakdown an $M \times N \times C \times D$ convolution layer (we can view this as the dimension of the convolution layer where $M \times N$ denotes the filter size, $C$ and $D$ represents the number of filters for input and output, respectively), into two convolution layers, consisting of a $1 \times 1 \times C \times D$ convolution layer followed by a $1 \times N \times C \times D$ convolution layer. This is similar to the $3 \times 3$ matrix = $(3 \times 1) \times (1 \times 3)$. Whereas the low complexity convolution layers used in both MobileNet [10], [11] and shuffleNet [12], [13] are mainly depthwise separable convolution layer [8], converting $M \times N \times C \times D$ into $C+ (M \times N \times 1 \times D)$ follow by $1 \times 1 \times C \times D$. In fact, $1 \times 1 \times C \times D$ is the development of [14] to improve cross-channel relations, initially termed as ‘mlpconv’. In addition, shuffleNet integrates low complexity convolution layers with the stochastic algorithm, which is channel shuffling, to improve model generalization at zero model size cost.

While the decomposition technique mentioned above works with the dimension of the convolution layers, SqueezeNet [15] and ResNet [16] leverage mlpconv to construct their ‘fire module’ and ‘bottleneck module’, respectively. In this context, a convolution module or block is defined as a combination of convolution layers and other algorithms described in Section III. Their proposed ideas are more towards utilizing mlpconv to expand or contract the number of output filters, corresponding to $D$ of $1 \times 1 \times C \times D$, without overly increasing the model size as the network gets deeper. In addition, this technique has also been used in CondenseNet [17] and ResNeXt [18]. Moreover, [15], [17], [18] adapted group convolution that divide the convolution layer operation channel-wise, corresponding to $C$ and $D$ of $M \times N \times C \times D$. Let $G$ denotes the factor to divide the convolution layer, the convolution layer will transform to $G + (M \times N \times C/G \times D/G)$.

It is challenging to distinguish which method of model compression is superior as other factors will affect the accuracy of the model, such as the CNN architecture design, its applications, and requirements. Parameter pruning is the most popular approach as it can be directly applied to an existing network and when transfer learning is highly desired.

Knowledge distillation on the other hand allows configuration on their networks (both teacher and student). However, they generally achieve less competitive performance as compared to pruning and convolutional factorization [4]. The other broadly used technique is quantisation, which can be easily implemented with other compression methods, as shown in Table 4. With a prior knowledge of CNN architecture design, one can have the flexibility to construct the CNN using convolution factorization. All the model compression discussed above can be combined to maximize the model performance.

To tackle acoustic scene classification with a restriction on the size complexity of the model, this paper approaches the problem by using quantization and convolution factorization as strategies. As parameter pruning and knowledge distillation are designed around having pre-trained models in mind. In particular, this paper heavily adapted MobileNet [10], [11] and shuffleNet [12], [13] as depthwise separable convolution layer has the lowest model size cost and the shuffle modules account to zero model size cost. This paper uses quantization to convert 32-bit floating points to 16-bit floating points.

Following the current state-of-the-art framework of combining time-frequency representation with CNN, this paper proposes using wavelet scattering (WS) [19]–[21] with a combination of MobileNet and ShuffleNet to tackle low-complexity computing for ASC. The combination of WS with CNN has already been well established [22]–[29].

In the design of CNN, [22], [24], [29] featured a two-stage CNN architecture design. A separate network trains the first and second-order coefficients in the first stage. In the second stage, both coefficients are concatenated together so that the network can learn the first and second-order as a single entity. For brevity, the term for first and second stage relates to CNN architecture design, while the term for first and second order relates to WS, the input representation of the CNN.
The reason for their setup, is due to the large disparity in magnitude between first and second orders. Their differences lie in how the second-order is being tackled. In the case of [22], they use a deep neural network (DNN) architecture, while [24], [29] uses a CNN architecture. In this context, DNN refers to stacks of fully-connected layers, or neural network will more than one hidden layer. Notably, the two-stage CNN architecture design used by [24],[ 29], is inspired by the work of [30], [31], who used it to split the log mel-spectrogram into low frequency and high frequency bins.

Yet in another application of a two-stage CNN architecture design, [20] utilized the first stage for training two features: WS and frequency scattering. Frequency scattering is introduced in [20] to make WS transposition invariance in frequency. Hence, based on the concept of applying log-mel-frequency spectogram with the discrete cosine transform (DCT) along the frequency axis to make mel-frequency cepstral coefficient (MFCC), they performed scattering transform of WS along the frequency axis. Overall, their design is not tailored towards low complexity modelling and does not address the limitation on the timescale. The reason why WS has a timescale limitation, even though wavelet transform provides multiscale based on the scaling functions in the theory of wavelets, is due to an averaging operation on the wavelet transform coefficients to make the representation translation invariant. The timescale that determines the size of the averaging function, limits the maximum scaling factor of the wavelet.

The scaling function is one of the most crucial attributes of wavelets. In the case of [32], they designed a ‘wavelet-based neural network’ that replaces the radial basis function of the neural network with scaling factor of the wavelet. They exploit the scaling factor as the learning function to improve network representation to be of an orthonormal basis. The studies of scaling properties have also been used in the image domain [33], [34]. The work of [33] involve the exploitation of discrete wavelet transform scaling properties to decompose synthetic aperture radar sec-ice texture providing multiscale/multiresolution. While [34] uses the concept of multi-scaling on the convolution filter size ($M \times N$) and dilation rate to create multiple feature maps with different receptive fields to improve reconstruction of super-resolution remote-sensing images. The dilation rate is a control for dilated convolution filter where when the dilation is 1, it is like a standard convolution filter. When the dilation rate increase to above 1, the size of the convolution filter ($M \times N$) will increase, but not the number of weights. Hence, dilation can be understood as the distance between each weight. Likewise, the concept of ‘multi-convolution filters’ is relatable to InceptionNet [9] design, where the problem they address is similar to Heisenberg Uncertainty Principle, but on convolution filter size.

Backtrack to ASC, [29] proposed multi-timescale by having a first order and three second-orders with different timescale set as the input representation for the first stage of the two-stage CNN. Next, they coupled it with genetic algorithm to reduce the large frequency dimension of all the second-order(s), which assist them in the reduction of computational time complexity.

However, for the application of low complexity model, ‘multi-timescaling’ is not a great option as additional resolution will result in the increase in model size complexity. Hence fore, instead of combining them together as exhibited in [29], this paper investigated a simple mixing of different timescales of the first and a single second-order to enrich the multi-timescale capability of WS. By coupling just the first and second-order together, the number of input representation in the first stage remain the similar to how WS is being handled, while providing a soft relief on the timescale limitation.

The ’mixed’ WS is then coupled with a low complexity CNN model, which has a similar structure to two-stage CNN architecture described in [24], but instead of residual convolutional block, this paper adapts MobileNet architecture, in addition to Shuffling modules. Notably, the major different between residual convolution block and mobileNet convolution block design, is the use of depthwise separate convolution layer instead of the standard $M \times N \times C \times D$ convolution layer. Hence, we present our CNN topology, a two-stage MobileNet architecture with shuffling modules, and we termed it two-stage mobile shuffling network (T-MSNet) for easy referencing.

In addition, we disputed that the current shuffling modules proposed by [12], [35] are not designed with time-frequency representation in mind. Taking advantage of the large frequency dimension of WS, we proposed a new shuffling module called ‘sub-spectral shuffle’ (SSS) that shuffles the feature maps ‘frequency-wised’. Notably, our experimental results show that SSS works best in the proposed setup of ‘mixed’ WS with low complexity CNN.

The remainder of the paper is structured as follows: Section II discusses wavelet scattering and described ‘mixed’ WS. Section III presented our proposed two-stage mobile shuffling network with further elaboration on our proposed sub-spectral shuffling modules. We then established the experimental setup using three different audio scene classification datasets in Section IV. In this paper, we perform our ablation study using DCASE 2021 Task 1a dataset and based on the corresponding experimental investigations, we evaluate the best ‘mixed’ WS and the best mobile shuffling networks on rest of the datasets. The discussion of the all the experimentation is presented in Section V, VI and VII. Lastly, to reiterate our contributions:

- We provide an evaluation on the ‘mixed’ WS and establish a discussion on the effectiveness of this approach.
- Following current ASC framework, we present our CNN topology, which combines IWS, MobileNet, Shuffling modules and two-stage CNN for low-complexity computing for ASC.
- Next, we proposed new shuffling modules called ‘sub-spectral shuffle’ (SSS) and ‘temporal shuffle’ (TS). Unlike channel shuffling (CS) and pixel-wise shuffling proposed by [12] and [35], respectively, our proposed shuffling modules are designed for time-frequency
representation. Their differentiation is described in Section III, C.

- We then demonstrated the effectiveness of the shuffling modules by evaluating it with 3 ASC datasets.

**FIGURE 1.** Multiple scaled ‘Morlet’ wavelet where the number of wavelet filters per octave, denoted as \( Q = 9 \).

**II. WAVELET SCATTERING**

In simplicity, wavelet scattering is constructed by cascading wavelet transform of a given signal [19]–[21]. The wavelet transform is a signal processing method that learns the modulation of the signal \( x \) through convolution operations of multiple scaled ‘mother’ wavelet, typically, the ‘Morlet’ wavelet. However, wavelet transform is not invariant to time-shift. There is a need to undergo non-linearity mapping through a complex modulus function \(| \ast |\) and an averaging function, making it locally translation invariant. Combining all these operations will produce the first-order coefficients [20]:

\[
S_1x (t, \lambda_1) = \mathcal{I} \ast x \ast \psi_{\lambda_1} \ast \phi(t)
\]  

where \( S_1x \) denotes the first-order coefficients, \( t \) is the time index, \( \psi_{\lambda_1} \) is a set of wavelet filters with \( \lambda_1 \) as the centre frequency and \( Q_1 \) is defined as the number of wavelet filters per octave. The set of wavelet filters is illustrated in Fig. 1. Lastly, \( \ast \) denotes convolution operation, and \( \phi \) is an averaging function, Gaussian filter and acts as a low-pass filter. While the averaging function ensures that the representation is locally invariant to time-shift smaller than a given timescale \( T \), it removes fine-scale information such as transient attacks. Hence, it becomes unstable to time-warping deformations. To be stable to deformation, it needs to satisfy the Lipschitz deformation stability condition [20], and wavelet scattering achieves this by calculating a second wavelet modulus transform on the first wavelet transform, \( |x \ast \psi_{\lambda_1}| \) based on [20]:

\[
S_2x (t, \lambda_1, \lambda_2) = \mathcal{I} \ast x \ast \psi_{\lambda_1} \ast \psi_{\lambda_2} \ast \phi(t).
\]  

Similar to first-order coefficients, \( S_2x \) denotes the second-order coefficients and the support of \( \psi_{\lambda_2} \) is centered in \( \lambda_2 \) with a frequency bandwidth of \( \lambda_2/Q_2 \) for \( \lambda_2 \geq 2\pi Q/T \) and \( 2\pi/T \) for \( \lambda_2 < 2\pi Q/T \). Following the suggestion of [19]–[35], this paper will only be using \( S_1x \) and \( S_2x \).

While WS does provide multiple timescales with a maximum scale of \( T \), it is still affected by Heisenberg Uncertainty Principle, where one does not know the range of time supports required to capture all the important spectra dynamics to better differentiate between one acoustic scene to another.

In addition, toggling different values of \( T \) will also affect the signal energy being captured between the first and second-order [19,20,21,b17]. A small \( T \) (e.g., \( \sim 46 \text{ms} \)) will have more energy saturated in the first order. In contrast, a larger \( T \) will see more energy dispersed to the second-order. The cause of energy dispersion is due to applying gaussian filter on the wavelet transform, \( \phi(t) \), with a scale of size \( T \) as expressed in (1) and (2). \( \phi(t) \), a low-pass filter of scattering coefficients, will result in a steeper slope when \( T \) is larger as illustrated in Fig. 2. Hence, more energy will be filtered out from the first-order and absorbed in the second-order where the coefficients are sparse. Insight of this analogy, this paper proposed a simple mixing of the first and second-order with different timescales. The idea is to bring diversity to the feature representation, and we termed it ‘Interleaving’ WS (IWS). This technique is effective for low complexity computing as the input representation has roughly the same dimension size as the usual WS while further enriched with multiscale. Relating to (1) and (2), WS and IWS can be
In subsection C, we proposed new shuffling modules called and provided the reason for using MobileNet in this context. We analysed the complexity of various convolution operations of T-MSNet. In subsection our CNN topology, we investigate the building blocks of being a low complexity model. With the establishment of T-MSNet and how it is designed to ensure that it hits the brief in subsection [12], [13] and two-stage CNN [24], which will be discussed based on the compilation of MobileNet [10], [11], ShuffleNet our proposed model T-MSNet. The construct of the CNN is represented of the CNN. In this section, we introduced we only evaluated the combination of IWS where timescale accuracy on DCASE 2021 dataset. Hence in Section V, A, activity. Based on the findings of [24], WS(s) with the timescale size of the input representation will increase model complexity. However, we faced a challenge whereby we must abide by the strict computational complexity constraint setup by DCASE 2021. Hence, the first stage only consists of a single convolutional layer to separately process S1X and S2X, with stride (1, 1) and (2, 1) respectively. Following the two-staged CNN architecture suggested by [24], we performed downsampling on S2X, with a stride of 2, on the Freq axis as aforementioned. This is to reduce biasness of the model on S2X caused by being a larger input representation (The larger the feature map the more convolution operations will be performed on the map, which will result in increased learning on the feature map). The second stage concatenates the features together to enable cross-spectral learning, which is further enriched when SSS is applied. The base architecture of T-MSNet after the first convolution layer follows a ‘lite’ version of mobileNet. The model has 8 stacks of MobileNet convolution blocks (MBloc) and the channels configuration for each convolution block is stacked as follows: \( \mathcal{C} = \{16, 24, 24, 32, 64, 64, 96\} \). Next, the shuffling modules, which include a combination of SSS, TS and CS, or in singularity, are slotted after the MBloc before each incremental of \( \mathcal{C} \) except for the first incrementation. Based on the findings of [35], there is a need to strategically positioned shuffling modules so that we can unleash their maximum potential.

Hence, the location of shuffling modules is predetermined at \( \mathcal{C} = \{16, 24, 24, S1, 32, 32, S2, 64, 64, S3, 96, S4\} \), where S1,S2,S3, & S4 are the indexed of their location as illustrated in Fig. 5, and Appendix. Table 6. While the location of the application of shuffling modules is set, it does not mean that shuffling will be applied. Inspired by [35], this paper analyzes the ‘activation’ of various combinations of shuffling modules at the predetermined location in Section VI. Finally, MSNet is completed with FCN described in subsection B and Table 1. Apart from the location of the shuffling modules, the number of segments for SSS, TS and CS are a set of hyperparameters. Based on our preliminary experiments, the configuration of the number of segments for SSS, TS, and CS should be different. For ease of referencing, we termed the number of segments for SSS, TS and CS as \( \mathcal{GBins}, \mathcal{GTim} \) and \( \mathcal{GChan} \), respectively. For the case of SSS, the number of segments used in S1 and S2 is more effective if it differs from S3 and S4.

We also found out that when \( \mathcal{GBins} \) is the same size as Freq, it will result in poorer classification results. Having \( \mathcal{GBins} = \mathcal{Freq} \) means that the entire spectral information is expressed as:

\[
WS = \begin{pmatrix}
S_1x(t, \lambda_1) \\
S_2x(t, \lambda_1, \lambda_2)
\end{pmatrix},
\]

\[
IWS = \begin{pmatrix}
S_1(x(t_1, \lambda_{11}) \\
S_2(x(t_2, \lambda_{21}, \lambda_{22})
\end{pmatrix}
\]

where \( t_1 \) is the timescale for the first order, and \( t_2 \) is the timescale for second-order. The construction of WS and IWS is illustrated in Fig. 3 and Fig. 4, respectively. Notably, there is a need to carefully limit the temporal dimension of IWS as the size of the input representation will increase model complexity. Based on the findings of [24], WS(s) with the timescale of \( T = \{92\text{ms}, 185\text{ms}, 371\text{ms}\} \) have the best classification accuracy on DCASE 2021 dataset. Hence in Section V, A, we only evaluated the combination of IWS where timescale \( T = \{92\text{ms}, 185\text{ms}, 371\text{ms}\} \).

III. TWO-STAGE MOBILE SHUFFLING NETWORK

Following the ASC framework, the IWS is set as the input representation of the CNN. In this section, we introduced our proposed model T-MSNet. The construct of the CNN is based on the compilation of MobileNet [10], [11], ShuffleNet [12], [13] and two-stage CNN [24], which will be discussed in subsection A and illustrated in Fig. 5. In addition to subsection A, we elaborate on the possible configuration for T-MSNet and how it is designed to ensure that it hits the brief of being a low complexity model. With the establishment of our CNN topology, we investigate the building blocks of T-MSNet. In subsection B, we examine the construct of MobileNet convolution block which design their convolution layers based on convolution factorization. In addition, we analysed the complexity of various convolution operations and provided the reason for using MobileNet in this context. In subsection C, we proposed new shuffling modules called sub-spectral shuffling (SSS) and Temporal shuffling (TS) and explained the intuition to our approach.

A. PROPOSED TWO-STAGE MOBILE SHUFFLE NETWORK ARCHITECTURE

The collective components of the MobileNet convolution block described in subsection B and shuffling modules described in subsection C present T-MSNet. In addition, T-MSNet featured a two-stage architecture design proposed by [24], [30], [31]. With [24] explaining that the two-stage approach is crucial in tackling the huge disparity in magnitude of the first and second-order coefficients. However, we faced a challenge whereby we must abide by the strict computational complexity constraint setup by DCASE 2021. Hence, the first stage only consists of a single convolutional layer to separately process \( S_1X \) and \( S_2X \), with stride (1, 1) and (2, 1) respectively. Following the two-staged CNN architecture suggested by [24], we performed downsampling on \( S_2X \), with a stride of 2, on the Freq axis as aforementioned. This is to reduce biasness of the model on \( S_2X \) caused by being a larger input representation (The larger the feature map the more convolution operations will be performed on the map, which will result in increased learning on the feature map). The second stage concatenates the features together to enable cross-spectral learning, which is further enriched when SSS is applied. The base architecture of T-MSNet after the first convolution layer follows a ‘lite’ version of mobileNet. The model has 8 stacks of MobileNet convolution blocks (MBloc) and the channels configuration for each convolution block is stacked as follows: \( \mathcal{C} = \{16, 24, 24, 32, 64, 64, 96\} \). Next, the shuffling modules, which include a combination of SSS, TS and CS, or in singularity, are slotted after the MBloc before each incremental of \( \mathcal{C} \) except for the first incrementation. Based on the findings of [35], there is a need to strategically positioned shuffling modules so that we can unleash their maximum potential.

Hence, the location of shuffling modules is predetermined at \( \mathcal{C} = \{16, 24, 24, S1, 32, 32, S2, 64, 64, S3, 96, S4\} \), where S1,S2,S3, & S4 are the indexed of their location as illustrated in Fig. 5, and Appendix. Table 6. While the location of the application of shuffling modules is set, it does not mean that shuffling will be applied. Inspired by [35], this paper analyzes the ‘activation’ of various combinations of shuffling modules at the predetermined location in Section VI. Finally, MSNet is completed with FCN described in subsection B and Table 1. Apart from the location of the shuffling modules, the number of segments for SSS, TS and CS are a set of hyperparameters. Based on our preliminary experiments, the configuration of the number of segments for SSS, TS, and CS should be different. For ease of referencing, we termed the number of segments for SSS, TS and CS as \( \mathcal{GBins}, \mathcal{GTim} \) and \( \mathcal{GChan} \), respectively. For the case of SSS, the number of segments used in S1 and S2 is more effective if it differs from S3 and S4.

We also found out that when \( \mathcal{GBins} \) is the same size as Freq, it will result in poorer classification results. Having \( \mathcal{GBins} = \mathcal{Freq} \) means that the entire spectral information is
being distorted. During our experimentation, we discovered that $GBins$ are best set at 10 for S1 and S2 and 5 for S3 and S4, for our configuration. As for TS, we set $GTim = Time$ as IWS has a small temporal dimension. Lastly, $GChan$ is most effective at 8, and the reason might be due to the number of channels of T-MSNet, which is relatively small to ensure the model size is within the given constraint.

The overview of the system includes the preprocessing to IWS and T-MSNet is illustrated in Fig. 5, and the T-MSNet architecture is described in Appendix, Table 6. With the understanding of the CNN topology, the following subsections will deconstruct the T-MSNet and discussed the convolution blocks and components.

**B. ADAPTING FROM MOBILE NETWORK CONVOLUTION BLOCK**

The MobileNet used in this paper is a combination of mobile network components designed by [10], [11], who carefully consider the extremely low computational cost. [10], [11] uses the concept of convolution factorization [8] to reduce the computational complexity significantly, and the adapted distinction can be summarized into three key components. The first component, which is the core convolution layer of MobileNet, uses depthwise separable convolution [8] to drastically reduce the computation in two steps. The first step focuses on capturing spatial relationships using a single filter per channel concept, called depthwise convolution, depicted in Fig. 6b. The second step employs a pointwise convolution as depthwise convolution does not capture channel-wise relationship. To better understand how depthwise separable convolution help in the reduction in model complexity, this paper provides a comparison analysis in Fig. 6 and compile a list of convolution layers and their computational analysis in Table 1. For ease of referencing, similar to Section I,
TABLE 1. Computational complexity of various convolution operations.

| No | Conv | Filter Dim | Parameters |
|----|------|------------|------------|
| 1  | Standard | 3 x 3 x 32 x 32 | 9,216 |
| 2  | Filter-wise | 3 x 1 x 32 x 32 + 1 x 3 x 32 x 32 | 6,144 |
| 3  | Depthwise | 3 x 3 x 3 x 32 | 288 |
| 4  | 1x1Conv | 1 x 1 x 32 x 32 | 1,024 |
| 5  | Group(4) | 3 x 3 x 8 x 8 x 4 | 2,304 |
| 6  | Depthwise separable | (3)+(4) | 1,312 |
| 7  | Shuffle | *Permutation | 0 |

Convolution Block

Convolutions Block

Classifier Layers

| No | Conv | Filter Dim | Parameters |
|----|------|------------|------------|
| 8  | VGCCC | 2 x (1) | 18,432 |
| 9  | 2 x MBloc | (4) = 1 x 1 x 32 x (32 x 6) | 2 x 8,896 = 17,792 |
| 11 | Flatten+ Dense | Flatten(30,4,96)+ Dense(96), Dense(96) | 1.1 (mil) |
| 12 | GAP+ Dense | GAP(30,4,96)+ Dense(96), Dense(96) | 19,594 |
| 13 | FCN | (4) = 1 x 1 x 96 x 96 | 10,176 |

Our Model

VOLUME 10, 2022

This table presents the computational complexity of various convolution operations discussed in Section III, A, based on the number of parameters. The number of parameters translates to the size of the model. The examples used in Row 1 to 9 reflect a convolution operation on a feature map with 32 channels, and the number of output channels is also 32. While for Row 11 to 13, we follow the output feature map size of our proposed model (Refer to Appendix, Table 6).

we defined a standard convolution layer with a tensor of $M \times N \times 1 \times D$ where $M \times N$ represents the size of the filters, $C$ is the number of filters, also called channels, and $D$ is the number of output filters.

Depthwise convolution layer is the first convolution layer of depthwise separable convolution design [8] and illustrated in Fig. 6b. Instead of filter-wise factorization, they break down the convolution layer channel-wise, such that $M \times N \times C \times D$ becomes $M \times N \times 1 \times D$ and a condition of $D = C$. This results in a considerable reduction in model size as $C$ is always larger than $M$ and $N$, especially when the network goes deeper.

The output of the depthwise convolution layer results from overly focused feature-wised learning and lacks understanding of the cross-channel relationship. Hence, a $1 \times 1 \times C \times D$ is employed to cover this semantic. In a depthwise separable convolution setup, $1 \times 1$ convolution $(1 \times 1$ Conv) is termed a pointwise convolution as it focuses on a single point of the feature map instead of a spatial relationship. As aforementioned, $1 \times 1$ Conv was designed by [14], and they intend to enhance the cross-channel relationship. The design of $1 \times 1$ Conv is also depicted in Fig. 6c. Lastly, the comparison between standard convolution against depthwise convolution and pointwise convolution is depicted in Fig. 6.

Hence, it is crucial to understand the computational complexity of these convolution operations as they have a direct impact to the size of the model and the calculation of the number of parameters is reflected in Table 1.

The second component incorporates an inverted residual and linear bottleneck structure [16], which signified the MobileNet convolution block design (MBloc) as illustrated in Fig. 7. The whole convolution block is defined by stacking expansion convolution, which is a $1 \times 1$ Conv followed by depthwise separable convolution, then a projection layer, which is also a $1 \times 1$ Conv. As the name suggests, the expansion convolution is designed to increase the number of feature maps. This elevates the effectiveness of depthwise separable convolution. The expanded channels provide a higher-dimensional feature space, leading to more expressiveness of the non-linear spatial-wised learning delivered by the depthwise convolution layer. The pointwise convolution condenses this information channel-wised into a sparse representation, maintaining a compact representation for the subsequent convolution operations [10,11]. The projection layer reflects the actual number of output channel and act as both a consolidator and a dimensionality reduction layer, producing a sparse set of feature maps.

Lastly, to reduce the computational cost caused by classifying with a fully connected layer, [36] proposed...
Before FCN is developed, the CNN model typically ends with a fully-connected layer(s) or dense layer(s) before the softmax layer. For this dense layer(s) to be effective, it requires a sizable number of hidden nodes, resulting in extremely high computational costs (e.g., $D^2$). Hence, [36] redesigned this portion by eliminating all dense layer(s) and instead of connecting the final convolution layer with a $1 \times 1$ Conv followed by a $1 \times 1 \times C \times E$, where $E$ is the number of classes. Global average pooling [14] is applied before the softmax layer. The result is a significant reduction in computational cost, as shown in Table 1.

Apart from the three components, to provide greater flexibility, mobileNet offers two hyperparameters to throttle the complexity, being ‘alpha’ and ‘expansion ratio’. The alpha value controls the “width” of the T-MSNet architecture, and as such, it affects the number of channels of every Mbloc. On the other hand, the expansion ratio is a hyperparameter setting for each Mbloc, and it controls only the size of the expansion layer. In this paper, based on our initial empirical study, we proposed the alpha to be 0.5 and all the expansion ratios to be 6.

### C. FROM CHANNEL AND SPATIAL SHUFFLE TO SUB-SPECTRAL AND TEMPORAL SHUFFLE FOR TIME-FREQUENCY REPRESENTATION

To further exploit IWS, this paper proposed a sub-spectral shuffling (SSS) module inspired by channel shuffle (CS) designed by [12] and spatial shuffle (SS) by [35]. Generally, shuffling is a stochastic approach and tends to improve the network performance without increasing the number of model parameters [35]. Hence, it is very applicable to designing a model with size restrictions. Shuffling channel-wise, pixel-wise or sub-spectral-wise encourages the filters to encompass information of different features. Notably, this paper also explored temporal shuffling (TS), which is the shuffling along the time axis. The difference between CS, SS, SSS, and TS is illustrated in Fig 8.

**FIGURE 8.** Variants of shuffle module with (a) being channel shuffle (CS), (b) being spatial shuffle (SS), (c) being sub-spectral shuffling (SSS), and lastly, (d) being temporal shuffling (TS). The feature maps represented in the figures have a dimension of $F \times T \times C$ (time-frequency representation with channels).
Given an input representation with a dimension size of \( \text{Freq} \times \text{Time} \times C \), where \( \text{Freq} \times \text{Time} \) represents the feature map which is a time-frequency representation, \( \text{Freq} \) being the frequency axis, and \( \text{Time} \) being the time axis. \( C \) is the number of feature maps and usually termed as a channel. CS performed shuffling on-axis \( C \), allowing cross-learning of feature maps. SS visualized feature map, \( \text{Freq} \times \text{Time} \), pixel-wise and randomly scrambled the individual pixels to destroy spatial memorization by the network. Thus, allowing the network to generalize more effectively. Although this makes sense for image classification, it has an adverse effect on acoustic scene classification. The poorer performance can be attributed to time-frequency representation, where a degree of temporal sequence and frequency locality are essential attributes to time-frequency representation, where a degree of temporal sequence and frequency locality are essential profiles. As such, this paper will present examples and explanations on SSS only. In addition, the algorithm is applicable to TS and CS.

Let \( GBins \) be the number of segments needed to segment the feature maps along the frequency axis, such that before SSS is being applied on the feature maps of \( \text{Freq} \times \text{Time} \times C \), the feature maps are grouped into bins of feature maps with \( \text{Freq/GBins} \times \text{Time} \times C \) dimension. During SSS operations, all the bins will undergo a random process of repositioning, as illustrated in Fig. 8. On top of that, it should also satisfy this condition \( 1 < GBins < \text{Freq} \). The combination of Mblok with shuffling modules is illustrated in Fig. 9d. Fig. 9 also present a comparison between our convolution block design against ShuffleNet and MobileNet.

### IV. EXPERIMENTAL SETUP

Kymatio [37] is used to process the raw waveform into IWS, and T-MSNet is implemented in TensorFlow version 2.1 [38]. All the models are trained and tested on a Window Desktop with Intel i7-11700F Processor, 32 GB of RAM and RTX 3090. The proposed model is being evaluated on three acoustic scene datasets. This is to investigate the effectiveness and robustness of IWS and SSS on various ASC tasks. The three datasets are described in subsection A. Subsection B provides details on the preprocessing, subsection C on the hyper-parameter setting, and subsection D describes the post-training quantization setup.

### A. DATASETS

The experiments are conducted on three widely used audio benchmark datasets – Environmental Sound Classification (ESC-50) [39], Urban Sound Dataset (US8K) [40], and Detection and Classification of Acoustic Scenes and Events (DCASE) [41] (DCASE 2021) presented in Table 2. Notably, these three datasets actually tackle two different classification tasks being, sound event recognition (SER) and what we now re-label as ASC. The ASC that we now know is as aforementioned the tasks to identify the vicinity we are at based on the sound produced in that scene. In this case, both ambient sound and event sound are important, though some sound events are noises for this task. As for SER, we are only interested in the sound event that happened in the scene.

### B. HYPERPARAMETERS

For all the experiments, this paper uses stochastic gradient descent (SGD) [42] for the learning strategy with a warm restart scheduled at a set of epoch indexes being \( (3, 7, 15, 31, 63, 126, 254) \). The learning rate will gradually decrease from 0.1 to a minimum limit of 0.00001 at each step based on half a cosine curve called ‘cosine annealing’ and restart back to 0.1 at each epoch index. The max epoch is set at 255 to experience 5 restarts with different intervals and a mini-batch size of 32. Notably, no data augmentation is being used for all the experimentation.

### C. POST-TRAINING QUANTIZATION

DCASE 2021 has an additional requirement: the model must be \( <= 128KB \), which translates to a model with 65536 parameters as our approach applies post-training quantization [7] that converts the T-MSNet from 32-bit to 16-bit floating points.

### D. WAVELET SCATTERING IMPLEMENTATION

In this paper, we proposed the use of WS and IWS to preprocess raw waveform to a time-frequency representation with different timescale \( T \) as discussed in Section II. It was observed that a small \( T < 92 \text{ms} \) is not effective for most acoustic classification tasks [20,22,b17-b22]. Hence, the minimum range of \( T \) is set to 92ms. In addition, based on the collective empirical studies, \( T \) provide the best result when set to an approximate of 300ms and is exhibited in Table 3. Therefore, in this paper, \( T = 92ms, 185ms, 371ms \). For the value of \( Q_1 \), [20] recommended to set it as \( 8 \), as it reflects a representation with similar attribute as mel-scale spectrogram, which is essential for environmental sounds classification. While yet another studies by [b19,b22], where they exhaustively evaluated a series of \( Q_1 \), has demonstrated

### TABLE 2. Details of the datasets.

| No | Dataset    | No of Data | #Classes | Duration (s) | Fold(s) |
|----|------------|------------|----------|--------------|---------|
| 1  | ESC-50     | 2000       | 50       | 5            | 5       |
| 2  | US8K       | 8732       | 10       | <=4          | 10      |
| 3  | DCASE 2021 | 23040      | 10       | 10           | 1       |

This table presents the details of the datasets used to evaluate T-MSNet. The column “Fold” describes the number of predefined cross-validation folds provided by the authors who curated the dataset. As DCASE 2021 has only provided a single training/testing setup, we iterated the setup five times and averaged the classification results.
that $Q_1$ is in fact optimal when set to 9. This has led us to a preliminary study of choosing the value of $Q_1$ between 8 and 9, and based on that, we have selected $Q_1$ as set to 9 for both $S_1x$ and $S_2x$. As for $Q_2$, [20], [22]–[28] is in unison of setting it to 1. Notably, $Q_2$ is only applicable to $S_2x$. [22], [24] also recommended the addition of the deltas and delta-deltas and concatenating them channel-wise, giving $C = 3$. Notably, the changes in the timescale, $T$, will affect the size of the Time dimension. For the concatenation operations (refer to the ‘plus’ sign of Fig. 5) to work, the feature maps of first and second orders need to share the same Time dimension size. Hence, we set the Time dimension to follow the timescale where $T = 185ms$, giving us a Time dimension size of 56. ‘Oversampling’, a feature in Kymatio (in Python) that increases the Time dimension, is being used when $T = 371ms$ to bridge the gap. We increase the strides by 2 for the time axis of the first convolution layer when $T = 92ms$.

V. INVESTIGATION ON IWS

In subsection A, we exhaustively search for the best combination of IWS using the DCASE 2021 dataset. In subsection B, we evaluate the effectiveness of IWS and WS with shuffling modules.

A. COMPARATIVE ANALYSIS OF IWS AGAINST WS

We exhaustively evaluated various combinations of IWS where the timescale $T_1, T_2 \in \{92ms, 185ms, 371ms\}$ and the result is featured in Table 3. Based on the root means square (RMS) of the combined accuracy of DCASE 2021, ESC-50 and US8K, mixing of the first and second-order based on this rule $T_1 < T_2$ shown to improve the classification performance of WS. Hence, our finding is in line with the analogy of energy dispersion of the first and second-order based on timescale discussed by [19], [20], [21], [23].

As scattering transform is contractive, [19], [20], [21], [23] calculated the energy of dispersion based on the fraction of the squared Euclidean norm of a vector of coefficients of the raw signal $x$ against both $S_1x$ and $S_2x$ (e.g., $\|S_1x\|^2 / \|x\|^2$).

Their calculation review that as $T$ get larger, there will be a shift of energy dispersion from first order to second order. Considering this analysis, we have plotted the log-scalogram of the first and second order of the sound of bird chirping in Fig. 10 and Fig. 11. Furthermore, in Section II, we plot the illustration of the gaussian filters in Fig. 2 to illustrate the reason for this phenomenon. $T$ affects the scale of the gaussian filter, which in this context is a low-pass filter. Hence, building on this concept, we proposed a simple mixing of first and second-order with different timescales which has demonstrated to be rather effective, especially for ESC-50 which consists of both SER and ASC recordings. While slight improvement is observed from the other two datasets. In addition, the logic of $T_1 < T_2$ does increase the number of wavelet transforms, especially for fine-scale
resolution. Figuratively, the increase of resolution of the second-order, as $T$ increase from 92ms to 371ms is by at least 100s over additional wavelet transform as illustrated in Fig. 10. This tentatively means an increase of wavelet filter’s variation, which result in capturing more acoustic characteristic as explained in Section II. More wavelets with different scales can reduce the effect of Heisenberg Uncertainty Principle. In contrast, setting $T_2 < T_1$ does not work well, typically, this pairing of IWS will always result in a poorer performance against their WS counterpart as reflected in Table 3, further backing the energy dispersion theorem.

However, a caveat appears when we start mixing first and second order with different timescales. There is a possible occurrence of feature redundancy or irrelevancy supported by the findings of [20,29]. The possibility of redundancy is further grounded by the energy dispersion theorem [19,20,21,b17], where scattering transform is contractive meaning,

$$\|S_0 x\|^2 + \|S_1 x\|^2 + \|S_2 x\|^2 + \ldots + \|S_m x\|^2 = \|x\|^2$$

for $m = \infty$. Hence, there will be a possibility of overlapping where the signal is captured by both first order and second order. Coupling with CNN that thrive in complex representation, it might result in building a stronger relationship between the overlapped wavelet scattering coefficients.

Thus, IWS can seem to be a double-edged sword without thorough experimentation, in this paper, we present the best combination of IWS for this setup is the combination of $T_1 = 185ms$ with $T_2 = 371ms$.

### B. EFFECTIVENESS OF SHUFFLING MODULES ON IWS AND WS

We then evaluate the combination of WS and IWS with one of the top shuffling modules presented in Table 3. The combination of SSS234, CS234 is being selected. Overall, the inclusion of shuffling modules has resulted in better performance for both WS and IWS. (Only one exception where $T_1 = 371ms$ with $T_2 = 185ms$) illustrated in Fig. 12 and Table 3. Our intuition to take advantage of the large frequency dimension of WS and IWS to perform SSS has proved to be fruitful. This finding is based on the consistent improvement of performance when $T_2 = 371ms$. In this setup, when $T_2 = 371ms$, the frequency dimension of WS and IWS is the largest, where $Freq = 724$, as illustrated in Fig. 11. Based on our preliminary study, for the following experiments, VI and VII, we use the IWS configuration where $T_1 = 185ms$with $T_2 = 371ms$.

### VI. ABLATION STUDY OF SHUFFLING MODULES

This section performed an ablation study of SSS, TS, and CS. Our ablation study is broken down into two parts: the first part is an evaluation of the individual type of shuffling modules. In this experiment setup, we explored the relationship between the position of the shuffling module and the number of shuffling modules that will influence the performance of the T-MSNet. The second part involves the study of the synergy of collectively combining SSS, TS, and CS. Lastly, we evaluate the robustness of shuffling modules on all the datasets presented in Section V, A.

#### A. ABLATION STUDY OF SSS, TS AND CS

The tabulated result is presented in Appendix, Table 7 and Table 8, classification accuracy and logloss, respectively.
Logloss is the measure of the confidence level of the model when making the prediction. (e.g., high confidence means that the logits on the ‘true’ label are nearing 1, and it has a logloss value approaching 0. Low confidence means that the logits on the ‘true’ label are far from 1, for example, 0.5, this will result in a higher logloss value) In this experiment, we investigated the effectiveness of the individual type of shuffling modules when ‘activated’ in different locations of T-MSNet, being S1, S2, S3 & S4 termed in Section 4, D.

Overall, SSS has the best performance in classification accuracy and logloss as illustrated in Fig. 13. This exhibits that SSS is more effective for a time-frequency representation, especially IWS (due to its high-frequency dimension). In fact, the first operation of SSS, which splits the feature maps ‘frequency-wise’, corresponds to the signal processing technique of applying equal frequency binning or applying linearly scaled filters on the spectrogram. (The more common practice is applying spectrogram with mel-scale filterbanks) Hence, a slice of the feature maps has similar characteristics to a frequency bin. Next, the shuffling operations mix the frequency bins around such that the feature maps are no longer ordered based on frequency magnitude (e.g., 0-100Hz, then 101-200Hz, then 201-300Hz, …). This forces the network to learn the general “shape” of the acoustic rather than the pitch. E.g., It no longer matters to the network whether it is a female voice or male voice since it just generalized that this is a speech. Hence, the general consistency in the improvement of logloss is demonstrated by SSS. The improvement of logloss is also reflected in other datasets. (Ref. Fig. 13b)

However, the effect of improved model generalization is lost when we apply SSS on S4. The inclusion of SSS on S4 has been demonstrated to result in a higher logloss. We attributed the cause of this observation to the relatively smaller dimensional size of Freq, which is 30. (Refer to Appendix Table 6) The poorer performance observed when shuffling with a low-dimension size is in unison with the findings [35]. They concurred that channel shuffling is ineffective at earlier layers because of limited channels. Another observation we made is that the classification accuracy dips when SSS is activated on S1. We deduced that the network needs to learn initial spatial relationships and to shuffle is better on feature maps with higher dimensional space. (a few layers of CNN with non-linear activation function)

The same setup is also being performed on TS and CS. However, their classification accuracy performances are erratic and do not improve the logloss. Furthermore, in some cases, the classification accuracy worsens with the inclusion of TS or CS. This observation is again supported by the findings of [35], where shuffling does not work effectively on low-dimension sizes.

Our proposed SSS works best on a low-complexity model with IWS with high dimension size of Freq. On top of that, the inclusion of a single SSS on S2 provides the best overall performance.

B. ABLATION STUDY OF COMBINING SSS, TS AND CS

While the study of subsection A provides an insight into the individual effectiveness of each shuffling module, this subsection explored the possible synergy when combining all of them. It is incredibly time-consuming to evaluate all the types of combinations. We first draw out some rules based on the observation from subsection A to reduce the number of possible tests. The rules are as follows:

- SSS must always be included as it provides significant improvement to logloss.
- There should not be a combination where SSS is solely activated in S1, S4 and S34 (S34 stands for SSS activated on both S3 and S4), due to poor logloss performance.
Based on the rules, our focus is now shifted to SSS being the lead shuffling module and how TS and CS can enhance it. Even with the rules, there are still several combinations. Hence, we performed a random search (more than 20 combinations, the results are presented in Appendix, Table 11). As no particular findings can be observed, this paper presents the top 3 combinations in Table 3. The overall performance of the combinations is relatively close. Hence, we evaluated other datasets to establish consistency and test the robustness of shuffling modules.

C. ROBUSTNESS OF SHUFFLING MODULES

In this subsection, we evaluated the robustness of shuffling modules with additional datasets described in Section IV, A. A slight progressive improvement is observed when we mixed first and second-order to form IWS for the US8K and DCASE 2021 datasets, while a significant improvement is observed for ESC-50. This can be attributed to the number of classes we need to classify. The variety of sounds recorded for ESC-50 is larger than US8K and DCASE 2021 (the number of classes is 5 times larger). Thus, it has a richer acoustic characteristic, and by mixing different timescales, the model can distinguish the different acoustic profiles better.

Yet another observation we notice when comparing the model performance on ESC-50 against US8K and DCASE 2021. A significant increase in performance is observed when IWS was applied, and further alleviated when shuffling modules are activated on ESC-50. This can be attributed to the size of the dataset. ESC-50 is a small dataset, and CNN is known for overfitting when the sample size is small. Hence, this finding reaffirmed that the shuffling algorithm could improve the generalization of the model, as mentioned by [12], [13], [35]. Another telltale of better generalization can be observed by the consistent improvement of logloss when shuffling modules are applied, as exhibited in Fig. 12b.

Overall, shuffling modules can improve classification accuracy and logloss performance for all the datasets. Generally, the top 3 T-MSNet configurations presented in Table 3 has very close performance rating. While the combination of SSS, TS and CS has been shown to provide the best classification accuracy, the activation of a single SSS offers a less complex model (by complex, we meant that it has fewer hyperparameters to configure). In summary, sub-spectral shuffling has been demonstrated to be the most effective shuffling module for our proposed system setup.

VII. COMPARATIVE ANALYSIS

In this section, we compared our proposed T-MSNet with SOTA(s) models presented in the various datasets.

A. ESC-50

Our proposed model has an advantage over human classification accuracy presented by [39]. L3-net developed by [43] uses self-supervised learning with an unlabeled video dataset to build audio and image embeddings. In the case of [41], they have demonstrated the effectiveness of embeddings on the DCASE 2021 dataset. However, the model size complexity cost of embeddings is too large. Hence, it is not recommended for low complexity modelling.

| Method | Accuracy/Logloss | Parameters/Size |
|--------|------------------|-----------------|
| ESC-50 | AudioSet: 81.80% Scratch: 74.77% | 635,002/- |
| CNN CF Quantization | 79% | ~508K/2.03MB |
| CNN of [46] Pruning Quantization | 70.93% | 130K/133.12 KB |
| * Same as ESC-50 | 79.76% | 64,788/126KB |
| * With shuffling | 81.00% | 64,788/126KB |

The table presents the consolidated SOTA of 3 datasets. * denotes our proposed model T-MSNet. Logloss is not recorded for ESC-50 and US8K as models are compared based on classification accuracy. AVC stands for Audio-visual correspondent, a self-supervised learning approach to build audio and image embeddings. End-to-end means that the network takes in the raw waveform as the input representation. CF stands for convolution factorization. KD stands for knowledge distillation. Generally, all the models are CNN.
While compared to [44], the addition of pretraining the model with a very large dataset, AudioSet [45], has improved performance dramatically. However, a lesser gain is observed when the same setup is applied on US8K. This can be attributed to the size of the dataset; transfer learning from a very large dataset is more effective when the dataset is small. The effect diminishes when applied to a larger dataset. Although transfer learning from a very large dataset has shown to be very useful, it requires massive resources and training time first to learn the semantics of the very large dataset.

Lastly, [46] proposed a model that has nearly the same model size as ours. Pruning and quantization are used to bring down the model’s parameters drastically. Our proposed model has a slight edge over the pruning technique in this scenario. Overall, the models used in ESC-50 are not geared towards low-complexity modelling. Hence, our best benchmark is against [39] and [46]. Notably, without the limitation on model size, [54], [55] are current SOTA models that achieve ~95% and ~97% classification accuracy, respectively. The common technique used by both models is the pre-learning of semantics from the external dataset(s). In the work of [54], they adapted ‘Transformer’ network and modified it to be suitable for audio classification. ImageNet and AudioSet are used to pretrain the network. As for [55], they incorporated a CLIP framework with a CNN model dedicated to audio classification. In short, the CLIP framework is the joined learning of multi modalities, typically visual and textual. They extended the number of modalities to include audio modalities. The output of CLIP is a shared multimodal embedding space. Similar to [43], [54], [55] requires tremendous resources and training time.

### B. US8K

Similar to ESC-50, US8K is not geared towards low-complexity modelling. Hence, we remain competitive against the models presented in Table 4. While comparing to [43], the inclusion of an additional dataset only shows a slight advantage against our model. In the work of [47], they uses convolution factorization and shares the same approach as our model. However, our model has the edge over their model in terms of both performance and model size. Our differentiation lies with the inclusion of shuffling modules, which has improved the generalization of our model. Lastly, [48] reuses the CNN model proposed by [46] and performs a series of pruning and quantization. Their best result that pushes the model size to 133.12KB only stands at 70.93%. In addition, [55] has also been evaluated with US8K and achieved a SOTA result of 90.07%.

### C. DCASE 2021

Lastly, Table 5 shows the comparison of the top 3 models against T-MSNet. The models are ranked based on DCASE 2021 Task1a challenge result. The main difference between our proposed model against theirs is the use of a parameters pruning strategy. Parameters pruning removes weights that the employed pruning strategy deemed less important. This strategy has proved to be extremely effective in compressing the size of a very large model [49]–[52], hence, the popularity of applying this technique. In addition to pruning, [49], [50] uses knowledge distillation to transfer ‘learning’ from the larger model to the pruned network.

Although our model falls short compared to logloss, our classification accuracy put us in third place. Furthermore, we present a different approach to carefully constructing our CNN architecture using convolution factorization and design patterns suggested by [10]–[13], [24], [35]. Instead of pruning and knowledge distillation, we proposed shuffling modules, a zero-model size cost algorithm, to improve model generalization.

Table 5 shows that pruning, convolution factorization, knowledge distillation, and quantization effectively compress weights in a standard convolution operation. The rest of the table are divided into Mblock(s), shuffling modules of either TS, SSS, CS or a combination of them, and the last layer is FCN. Mblock is represented by 1x1Conv, Depthwise convolution (DConv) and pointwise convolution (PConv). The BN and ReLU for Mblock are omitted from this table as they are illustrated in Fig. 6.

| TABLE 6. T-MSNet architecture design. |
|----------------------------------------|
| Ops | Stride | Filter Shape | Input Size |
| Conv | (1,1), (2,1) | 3 x 3 x 16 | 105 x 56 x 3, 724 x 56 x 3, 105 x 56 x 16, 362 x 56 x 16 |
| Concat+BN | | | |
| 1x1Conv | (1,1) | 1 x 1 x 16 | 467 x 56 x 16 |
| DConv | (1,1) | 3 x 3 x 16 | 467 x 56 x 16 |
| PConv | (1,1) | 1 x 1 x 8 | 467 x 56 x 16 |
| 1x1Conv | (1,1) | 1 x 1 x 48 | 234 x 28 x 16 |
| DConv | (1,1) | 3 x 3 x 96 | 234 x 28 x 96 |
| PConv | (1,1) | 1 x 1 x 16 | 234 x 28 x 96 |
| S1 | | | 234 x 28 x 16 |
| 1x1Conv | (1,1) | 1 x 1 x 96 | 117 x 14 x 96 |
| DConv | (2,2) | 3 x 3 x 96 | 117 x 14 x 96 |
| PConv | (1,1) | 1 x 1 x 16 | 59 x 7 x 96 |
| 1x1Conv | (1,1) | 1 x 1 x 192 | 59 x 7 x 32 |
| DConv | (1,1) | 3 x 3 x 192 | 59 x 7 x 192 |
| PConv | (1,1) | 1 x 1 x 32 | 59 x 7 x 192 |
| S3 | | | 59 x 7 x 32 |
| 1x1Conv | (1,1) | 1 x 1 x 192 | 59 x 7 x 32 |
| DConv | (2,2) | 3 x 3 x 192 | 59 x 7 x 192 |
| PConv | (1,1) | 1 x 1 x 48 | 30 x 4 x 192 |
| S4 | | | 30 x 4 x 48 |
| 1x1Conv+ | (1,1) | 1 x 1 x 96 | 30 x 4 x 48 |
| BN+ReLU | | | |
| 1x1Conv | (1,1) | 1 x 1 x 10 | 30 x 4 x 96 |
| GAP | | | 30 x 4 x 10 |
| Softmax | | | 10 |

This table presents the T-MSNet architecture. The first convolution layer is a standard convolution operation. The rest of the table are divided into Mblock(s), shuffling modules of either TS, SSS, CS or a combination of them, and the last layer is FCN. Mblock is represented by 1x1Conv, Depthwise convolution (DConv) and pointwise convolution (PConv). The BN and ReLU for Mblock are omitted from this table as they are illustrated in Fig. 6.
the model. However, supporting pruning and knowledge distillation is the requirement of a well-constructed CNN. In contrast, convolution factorization requires careful planning for each layer, and hence, both have their disadvantages. In addition to the above-stated techniques, this paper proposed shuffling modules. This stochastic method improves model generalization and favors low-complexity modelling as it does not constitute model size. Generally, our proposed model is competitive.

**VIII. CONCLUSION**

In this paper, we have proposed a mixing of WS called IWS and two kinds of shuffle: sub-spectral shuffling (SSS) and temporal shuffling (TS). Notably, shuffling modules are a ‘zero model size’ operation that helps to improve the model’s ability to generalize. We then demonstrated the modularity of shuffling modules and how they can be paired with MobileNet architecture. The final product is our proposed T-MSNet. Based on our experimentation, sub-spectral shuffling is the most effective shuffling module and even comes toe-to-toe with the combined shuffling modules. SSS vs top 2 combined shuffling modules; (80.9% vs 80.6%, 82.15%), (80.5% vs 80.3%, 81%) and (70.5% vs 70.4%, 70.6%) on ESC-50, US8K and DCASE 2021, respectively. Further research can explore better ways to finetune the hyperparameters of T-MSNet, pairing shuffling modules with other convolution neural network architecture, pretrain with Audioset. Apart from tuning the networks or applying it on other CNN architectures, we can also extend the study on other audio classification tasks, such as bioacoustics classification tasks or music auto-tagging. In addition, the time and

### Table 7. Classification result of various ‘activation’ of SSS, TS and CS, separately.

| No | Layer index | SSS | TS   | CS   |
|----|-------------|-----|------|------|
| 1  | 0           | 67.78 |      |      |
| 2  | 1           | 68.55 | 67.52 | 68.40 |
| 3  | 2           | **70.50** | 67.92 | 68.05 |
| 4  | 3           | 70.25 | 67.99 | 67.83 |
| 5  | 4           | 68.38 | 67.73 | 68.75 |
| 6  | 12          | 68.87 | 67.49 | 67.80 |
| 7  | 23          | 69.40 | 68.38 | 68.41 |
| 8  | 34          | 69.93 | 69.02 | 67.99 |
| 9  | 123         | 67.19 | 68.59 | 67.81 |
| 10 | 234         | 69.57 | 67.82 | 68.77 |
| 11 | 1234        | 67.79 | 68.25 | 67.96 |

This table reflects the classification result presented in Fig. 11a.

### Table 8. Logloss result of various ‘activation’ of SSS, TS and CS, separately.

| No | Layer index | SSS | TS   | CS   |
|----|-------------|-----|------|------|
| 1  | 0           | 1.447 |      |      |
| 2  | 1           | 1.246 | 1.435 | 1.445 |
| 3  | 2           | 1.185 | 1.432 | 1.452 |
| 4  | 3           | 1.273 | 1.449 | 1.445 |
| 5  | 4           | 1.450 | 1.448 | 1.456 |
| 6  | 12          | 1.189 | 1.427 | 1.467 |
| 7  | 23          | **1.162** | 1.422 | 1.448 |
| 8  | 34          | 1.293 | 1.427 | 1.462 |
| 9  | 123         | 1.188 | 1.416 | 1.447 |
| 10 | 234         | 1.170 | 1.428 | 1.452 |
| 11 | 1234        | 1.179 | 1.423 | 1.450 |

This table reflects the logloss result presented in Fig. 11b.

### Table 9. Classification result of shuffling modules on other datasets.

| No | Model | ESC-50 | US8K | DCASE 2021 |
|----|-------|--------|------|------------|
| 1  | WS    | 68.4   | 78.39 | 67.47      |
| 2  | IWS   | 73.75  | 79.76 | 67.78      |
| 3  | SSS2  | 80.90  | 80.94 | 70.58      |
| 4  | SSS234,CS234 | 80.60  | 80.26 | 70.39      |
| 5  | SSS2,CS3,TS34 | **82.15**  | **81.00** | **70.60**      |

This table presents the classification result presented in Fig. 12a.

### Table 10. Logloss result of shuffling modules on other datasets.

| No | Model | ESC-50 | US8K | DCASE 2021 |
|----|-------|--------|------|------------|
| 1  | WS    | 2.044  | 1.161 | 1.425      |
| 2  | IWS   | 1.946  | 1.172 | 1.447      |
| 3  | SSS2  | 1.036  | **0.910** | 1.185 |
| 4  | SSS234,CS234 | 1.025  | 0.922 | **1.148** |
| 5  | SSS2,CS3,TS34 | **0.965**  | 0.911 | **1.152** |

This table presents the logloss result presented in Fig. 12b.

### Table 11. Result of various combination of SSS, TS and CS.

| No | SSS | TS | CS | Accuracy | Logloss |
|----|-----|----|----|---------|---------|
| 1  | 4   | 0  | 4  | 67.67   | 1.437   |
| 2  | 34  | 0  | 34 | 68.90   | 1.292   |
| 3  | 0   | 4  | 234| 67.75   | 1.375   |
| 4  | 2   | 34 | 3  | **70.60** | 1.162   |
| 5  | 234 | 0  | 234| 70.39   | **1.148** |
| 6  | 2   | 34 | 0  | 70.24   | 1.171   |
| 7  | 2   | 34 | 0  | 70.15   | 1.159   |
| 8  | 2   | 34 | 0  | 70.15   | 1.179   |
| 9  | 2   | 34 | 0  | 70.12   | 1.181   |
| 10 | 2   | 34 | 0  | 70.00   | 1.149   |
| 11 | 234 | 0  | 1234| 69.88   | 1.163   |
| 12 | 23  | 0  | 234| 69.84   | 1.158   |
| 13 | 23  | 0  | 234| 69.77   | 1.189   |
| 14 | 2   | 0  | 234| 69.52   | 1.168   |
| 15 | 23  | 0  | 4  | 69.52   | 1.176   |
| 16 | 2   | 0  | 4  | 69.52   | 1.176   |
| 17 | 2   | 234| 3  | 69.50   | 1.166   |
| 18 | 2   | 0  | 24 | 69.48   | 1.161   |
| 19 | 234 | 0  | 34 | 69.14   | 1.163   |
| 20 | 234 | 0  | 34 | 69.14   | 1.163   |
| 21 | 234 | 0  | 34 | 68.87   | 1.162   |
| 22 | 1234| 0  | 1234| 67.66   | 1.174   |
| 23 | 1234| 0  | 1234| 67.43   | 1.181   |

This table presents 20 different combinations from random search. The first examples demonstrated the performances result if we do not follow the rules we setup in Section VI B. In particular, their logloss result is not competitive. The rest of the table is then arranged based on classification accuracy in descending order (meaning the top performance models will be placed at the top of the table).
frequency shuffling modules can also be applied on other time-frequency representations, such as spectrogram. While looking internally at WS algorithm, we can explore other type of ‘mother’ wavelet such as “Mexican Hat Wavelet” or “Meyer Wavelet”.

APPENDIX
See Tables 6–11.
I. Martín-Morató, A. Mesaros, and T. Virtanen, “Acoustic scene classification in DCASE 2020 challenge: Generalization across devices and low complexity solutions,” in Proc. Detection Classification Acoustic Scenes Events Workshop (DCASE), 2020, pp. 56–60.

I. Loshchilov and F. Hutter, “SGDR: Stochastic gradient descent with restarts,” CoRR, vol. abs/1608.03983, pp. 1–16, Aug. 2016.

R. Arandjelovic and A. Zisserman, “Look, listen and learn,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 609–617.

P. Lopez-Meyer, J. A. del Hoyo Ontiveros, H. Lu, and G. Stemmer, “Efficient end-to-end audio embeddings generation for audio classification on target applications,” in Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP), Jun. 2021, pp. 601–605, doi: 10.1109/ICASSP59728.2021.9414229.

J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, “Audio set: An ontology and human-labeled dataset for audio events,” in Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP), Mar. 2017, pp. 776–780.

M. Mohaimennuzzaman, C. Bergmeir, I. T. West, and B. Meyer, “Environmetal sound classification on the edge: A pipeline for deep acoustic networks on extremely resource-constrained devices,” 2021, arXiv:2103.03483.

X. Zhang, Y. Zou, and W. Wang, “LD-CNN: A lightweight dilated convolutional neural network for environmental sound classification,” in Proc. 24th Int. Conf. Pattern Recognit. (ICPR), Aug. 2018, pp. 373–378.

M. Mohaimennuzzaman, C. Bergmeir, and B. Meyer, “Pruning vs XNOR-net: A comprehensive study of deep learning for audio classification on edge-devices,” IEEE Access, vol. 10, pp. 6696–6707, 2022.

B. Kim, S. Yang, J. Kim, and S. Chang, “QTI submission to DCASE 2021: Residual normalization for device-imbalanced acoustic scene classification with efficient design,” Detection Classification Acoustic Scenes Events (DCASE), Tampere, Finland, Tech. Rep. DCASE2021_Kim_36_t1, 2021.

C. H. Yang, H. Hu, S. M. Siniscalchi, Q. Wang, Y. Wang, X. Xia, Y. Zhao, Y. Wu, Y. Wang, J. Du, and C. Lee, “A lottery ticket hypothesis framework for low-complexity device-robust neural acoustic scene classification,” Detection Classification Acoustic Scenes Events (DCASE), Tampere, Finland, Tech. Rep. DCASE2021_Yang_124_t1, 2021.

K. Koutini, J. Schluter, and G. Widmer, “CPJKU submission to DCASE21: Cross-device audio scene classification with wide sparse frequency-damped CNNs,” Detection Classification Acoustic Scenes Events (DCASE), Tampere, Finland, Tech. Rep. DCASE2021_Koutini_112_t1, 2021.

I. Martín-Morató, T. Heittola, A. Mesaros, and T. Virtanen, “Low-complexity acoustic scene classification for multi-device audio: Analysis of DCASE 2021 challenge problems,” 2021, arXiv:2105.13734.

J. Frankle and M. Carbin, “The lottery ticket hypothesis: Finding sparse, trainable neural networks,” 2018, arXiv:1803.03635.

Y. Gong, Y.-A. Chung, and J. Glass, “AST: Audio spectrogram transformer,” in Proc. Interspeech, Aug. 2021, pp. 571–575, doi: 10.21437/Interspeech.2021-698.

A. Guzhov, F. Raue, J. Hees, and A. Dengel, “AudioCLIP: Extending CLIP to image, text and audio,” 2021, arXiv:2106.13043.

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

From 2011 to 2015, he was a Software Engineer. From 2015 to 2017, he was a Team Lead/a Technical Lead, and from 2017 to 2018, he was 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 (Senior Member, IEEE) received the B.Eng. degree (Hons.) in mechanical and aerospace engineering from Nanyang Technological University (NTU), in 2000, the M.Sc. degree (Hons.) in advanced control and systems engineering from The University of Manchester, in 2001, and the Ph.D. degree in applied control engineering from the Research Robotics Centre, NTU, in 2008.

He is currently an Associate Professor (a Reader) at Newcastle University, 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 three U.S. patents in the area of electronics test systems and components. He is the author/coauthor of over 100 peer-reviewed papers on journals and conference proceedings. His research interests include designing and simulating complex systems in an uncertain environment, such as marine electric vehicles, energy storage systems, and acoustic systems.

Dr. Chin is an Associate Editor of IEEE ACCESS and IEEE TRANSPORTATION ELECTRIFICATION COMMUNITY (TEC) eNewsletter. He has served as the general chair and the publicity chair in various international conferences.

YE LI received the master’s degree in mechatronics from the University of Southern Denmark, in 2009, and the Ph.D. degree in biomedical engineering from the 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, and building applications for the Internet of Things (IoT). His primary focuses are sensor fusion, anomaly detection, and time-series analysis.