Pre-trained DenseNet-121 with Multilayer Perceptron for Acoustic Event Classification

Pooi Shiang Tan, Kian Ming Lim, *Member, IAENG,* Cheah Heng Tan, Chin Poo Lee, *Member, IAENG,*

***Abstract*—Acoustic event classification aims to classify the acoustic event into the correct classes, which is beneficial in surveillance, multimedia information retrieval, and smart cities. The main challenges of acoustic event classification are insufficient data to learn a good model and varying lengths of the acoustic input signal. In this paper, a deep learning architecture, namely: Pre-trained DenseNet-121 with Multilayer Perceptron is proposed in this work to classify the acoustic events into correct classes. To mitigate the data scarcity problem, two data augmentation techniques: time stretching and pitch shifting, are applied on training data to boost the number of training samples. Given the augmented acoustic signal, a frequency spectrogram technique is then employed to represent the acoustic event signal into a fixed-size image representation. The output of the spectrogram images are enriched with the information of the acoustic signal such as energy levels over time domain, frequency changes, signal strength, and amplitude. Subsequently, a pre-trained DenseNet-121 model is adopted as a transfer learning technique to extract significant features from the spectrogram image. In doing so, computation resources can be greatly reduced and improve the performance of the deep learning-based model. Three benchmark datasets: (1) Soundscapes1, (2) Soundscapes2, and (3) UrbanSound8K, are used to assess the performance of the proposed method. From the experimental results, the proposed Pre-trained DenseNet-121 with Multilayer Perceptron outperforms existing works on Soundscapes1, Soundscapes2, and UrbanSound8K datasets with the F1- scores of 80.7%, 87.3%, and 69.6%, respectively.**

***Index Terms*—acoustic event classification, DenseNet, multilayer perceptron, time stretching, pitch shifting, frequency spectrogram**

I. Introduction

ACOUSTIC event classification (AEC) is a task to correctly classify acoustic signals that contain acoustic events into the actual event classes. AEC is widely implemented in automatic sound segmentation, home living monitoring systems, and surveillance systems. However, AEC remains challenging due to the variation in recording background, varying of events length, insufficient training samples, and etc. In view of this, techniques to analyse and extract meaningful features from the acoustic signals are important in AEC tasks. An acoustic signal that contains acoustic events can be treated as a temporal data, which is a sequence of samples ordered in their occurrence over time domain. Generally, there are two major approaches in AEC: (1) hand-crafted approaches, and (2) deep learning-based approaches.

For the hand-crafted approaches, robust features such as Mel Frequency Cepstrum Coefficients (MFCC) are extracted from the acoustic event signal. Thereafter, robust classifiers, for instance, Support Vector Machine (SVM) are used to classify the feature vector into the correct classes. However, a single feature extraction technique is insufficient to achieve state-of-the-art performance in AEC. Hence, it leads to aggregation of multiple acoustic features to obtain better representation. By doing so, the computation complexity is increased and it is undesired for real-time implementation. On the contrary, deep learning-based approaches attempt to build an AEC model in an end-to-end manner. Deep learning has been widely applied in many applications [1]. Throughout the training process, deep learning models are able to automatically discover a set of significant representations instead of manually engineered features that lack robustness to unseen data. In view of this, we propose a deep learning-based acoustic event classification model, referred to as Pre-trained DenseNet-121 with Multilayer Perceptron (MLP).

In the proposed Pre-trained DenseNet-121 with MLP method, two data augmentation techniques: (1) time stretching, and (2) pitch shifting; are employed in the training set in order to overcome the data scarcity. Current work in AEC faces the difficulties to mitigate the problem of varying lengths of acoustic data. In view of this, instead of processing the temporal acoustic data, an effective conversion strategy is proposed to convert the acoustic signal into frequency spectrogram which enriches with more information of the acoustic signal such as strength of the signal, frequency changes, and energy levels over the time domain. Moreover, the colour or brightness in a spectrogram represents the amplitude of an acoustic signal. The problem of varying length of the acoustic signal is solved automatically during the conversion process by fixing the dimension of the spectrogram image. Thereafter, to obtain a compact feature representation from the spectrogram, a transfer learning with pre-trained DenseNet-121 architecture is proposed. The DenseNet-121 model used in this work is pre-trained on the ImageNet dataset. Instead of training from scratch, the pre-trained DenseNet-121 model is utilised to extract robust features with the transferred knowledge. This will further enhance the performance of the proposed model and it also saves the training time.

Given the compact features obtained through transfer learning with the pre-trained DenseNet-121 model, the Multi-

Manuscript received February 20, 2022; revised September 22, 2022. This research is supported by Telekom Malaysia Research & Development under grant number RDTC/221045 and Collaborative Research in Engineering Science and Technology (CREST) R&D grant No. CREST/R&D/P07C2-17/005.

P.S. Tan is a postgraduate student in the Faculty of Information Science and Technology, Multimedia University, Melaka, 75450 Malaysia. (e-mail: 1111115046@student.mmu.edu.my)

K.M. Lim, the corresponding author, is a Senior Lecturer in the Faculty of Information Science and Technology, Multimedia University, Melaka, 75450 Malaysia. (e-mail: kmlim@mmu.edu.my)

C.H. Tan is a Distinguished Member of Technical Staff at Motorola Solutions Malaysia Sdn. Bhd., 2A, Medan Bayan Lepas, Bayan Lepas Innoplex, 11900 Bayan Lepas, Penang, Malaysia. (e-mail: cheah-heng.tan@motorolasolutions.com)

C.P. Lee, is a Senior Lecturer in the Faculty of Information Science and Technology, Multimedia University, Melaka, 75450 Malaysia. (e-mail: cplee@mmu.edu.my)

layer Perceptron (MLP) architecture is used to perform learning on the features extracted by the pre-trained DenseNet-121. MLP employs a supervised learning approach known as back-propagation where the weights are adapted to reduce the error rate during the training process. This allows the MLP to learn the complex and non-linear relationships between inputs and outputs. In addition, the trained MLP model is able to generalise to the unseen data. The performance of the proposed Pre-trained DenseNet-121 with MLP model is then evaluated on three acoustic event datasets.

The major contributions of this work are:

- Alleviate the impact induced by data scarcity via two data augmentation techniques: (1) time stretching, and (2) pitch shifting.
- Mitigate the varying lengths of acoustic signal by converting the input acoustic data into a fixed length spectrogram.
- An acoustic event classification model based on pre-trained DenseNet-121 with multilayer perceptron.

This paper is structured as follows: Section II reviews the current acoustic event classification techniques. Section III explains the proposed pre-trained DenseNet-121 with multilayer perceptron in detail. Section IV presents the experimental results and analysis. Lastly, the conclusion is drawn in Section V.

II. Related Work

In general, acoustic event classification can be grouped into hand-crafted approaches and deep learning-based approaches.

In the hand-crafted approach, the acoustic event classification consists of feature extraction and classification stages. The robust features in the input data are selected in the feature extraction stage. The dimensionality of the data is greatly reduced through the feature selection process. In the classification stage, the extracted feature vector is taken as the input to perform the classification. In the hand-crafted approach, the features are manually engineered to deal with particular tasks which cause limitations to adapt to non-identical challenges. [2] proposed an exemplar-based model to model the acoustic event as a linear combination of dictionary atoms. [3] proposed an acoustic event classification model that integrated Filter Back Coefficients (FC) and non-negative matrix factorisation (NMF). FC was used to seize the dynamic structure in the short-term features and NMF was used to learn the temporal filters. In [4], a combination of Gaussian Mixture Models (GMM) and Mel Frequency Cepstral Coefficients (MFCCs) was proposed to classify the acoustic data. Additionally, the proposed method utilised a shared background model to minimise the impact of silence in the acoustic data. Classification of the acoustic data was then determined based on the GMM model that produced the highest averaged posteriori score. Another work characterised by the hidden Markov model (HMM) [5] utilised MFCC as the feature extraction mechanism. Three states continuous-density HMM was used to model the sound-event-conditional feature. A mixture of multivariate Gaussian density functions was utilised to model the probability density functions in each state. In [6], MFCC was deployed as the feature selection unit and Support Vector Machine (SVM) was leveraged as the classifier. In their work, 13 MFCCs features were extracted for each audio file and a sliding window of 10 frames with 5 overlapping frames was used. For classification, slack SVM was employed. Later, a tandem connectionist-HMM [7] was introduced to integrate both sequence modelling capabilities of HMM and trained context-dependent discriminative capabilities of a neural network. To calculate the KL-divergence between the audio segments, an SVM-GMM-supervector was deployed. In [8], an acoustic event classification system that implemented extraction of spectro-temporal features and two-layer HMM was proposed. Noise reduction algorithms depend on log-spectral amplitude estimator [9] and noise power density estimation [10] were used in their work. Later on, some works [11], [12] applied random forest for acoustic event classification. In [11], an architecture that utilised long contextual information, low-dimensional discriminant global bottleneck features and category-specific bottleneck features, were proposed for acoustic event classification. Both global and category-specific bottleneck features were able to extract the preliminary knowledge of the acoustic event class. Then, the acoustic features were concatenated, and category-specific bottleneck features were passed into a random forest classifier. [12] decomposed the acoustic signal into several superframes that were annotated with event class labels. Random forest was then used to perform learning with the displacement knowledge.

Although hand-crafted approaches perform well in acoustic event classification, the hand-crafted features are not robust to different datasets or unseen data. Motivated by the successes of deep learning in computer vision [13]–[15], researchers had begun to use deep learning approaches in the acoustic event classification tasks [16]. Deep learning architecture is a deep neural network that comprises input layer, hidden layers and an output layer. The hidden layers in the deep learning architecture possess the representation learning ability. In [17], the acoustic data is converted into the energy-augmented spectrogram image features, and then fed into a 1-max pooling CNN (1MaxCNN-E-MC) for training and classification. In [18], a deep learning-based acoustic event classification architecture was introduced to solve polyphonic acoustic event classification tasks. Due to the limitation of the fixed threshold approach, the authors proposed the contour-based and regressor-based dynamic threshold schemes. They adopted a fully connected deep neural network (DNN) as the classifier. [19] adopted a DNN to extract robust features from the raw acoustic data. In addition, transfer learning techniques were applied in their work where the DNN filters were pre-trained in the source realm and fine-tuned to the target realm. After training, the trained DNN filters were used to extract features from the acoustic signal. [20] proposed a weakly-supervised learning framework, known as FrameCNN, for acoustic event classification. FrameCNN model received the 128-bin log mel-spectrogram as the input of the CNN to reduce the dimensionality along the frequency axis. A global temporal pooling layer was added to concatenate three pooling outputs with mean, maximum, and variance function. Moreover, a transposed convolution network was employed to perform frame-wise classification by training on weakly annotated data. Another work introduced by [21] used SB-CNN, which was a CNN based acoustic event classification model that comprises 3 convolutional layers, interleaved with 2 pooling layers, and followed by 2 dense layers. A feature vector of 193 values of five different features: Mel-frequency cepstral coefficients, Chromagram of a short-time Fourier transform, Mel-scaled power spectrogram, Octave-based spectral contrast [22], and Tonnetz were fed into SB-CNN for training. [23] proposed a CNN-based acoustic event classification model known as fine-resolution convolutional neural network (FRCNN) that utilised lateral construction to produce fine-resolution feature maps. Apart from that, depth-wise separable convolution was applied to decrease the number of trainable parameters. In [24], an acoustic event classification model that optimised frequency sub-band separation via CNN was proposed. Both damping ratios for roll off and cut-off frequencies were taken as the additional parameters to the CNN model. The filter frequency response was optimised for constructing salient features.

The capability of deep learning approaches for automatic feature engineering is one of the main advantages over hand-crafted approaches. Using deep learning, the input acoustic data will be analysed to discover the set of robust features that are correlated. The robust features are then integrated to become a feature map to accelerate the learning process. In view of this, we focus on deep learning based architecture for acoustic event classification in this work.

III. Pre-trained DenseNet-121 with Multilayer Perceptron

This section describes the proposed pre-trained DenseNet-121 with multilayer perceptron model in detail. The architecture of the proposed pre-trained DenseNet-121 with MLP is depicted in Fig. 1.

The training of deep neural networks is more susceptible to overfitting when there are limited training samples. In view of this, two data augmentation techniques are performed to boost the training samples and to prevent the overfitting. The two data augmentation techniques are pitch shifting and time stretching. Time stretching augments the training samples by modifying the speed of the acoustic signal, whilst pitch shifting produces more training samples by altering the pitch of the acoustic signal. To overcome varying lengths of the acoustic signals, the augmented acoustic signals then undergo conversion from time domain into frequency domain via frequency spectrogram technique. To obtain a set of significant features, transfer learning is employed. The output frequency spectrogram image is fed into a pre-trained DenseNet-121 which is adopted as a feature extraction mechanism to extract robust features from the frequency spectrogram. The extracted features from pre-trained DenseNet-121 are more diversified and it includes the features from all complexity levels [25]. The resulting significant features extracted from the frequency spectrogram are then used as an input stream for acoustic event classification. In the model stage, a multilayer perceptron is proposed to perform the training and classification of the output features from the pre-trained DenseNet-121 into the correct acoustic event classes.

A. Data Augmentation

Typically, deep neural networks require substantial amounts of training samples in order to achieve good performance. Data augmentation technique is used to overcome the data scarcity problem via increasing the number of the training samples by modifying the available data. In this work, two data augmentation techniques are employed to generate more training samples: (1) time stretching, and (2) pitch shifting. An illustration of the data augmentation techniques is presented in Fig. 2.

1) Time Stretching: Time stretching is a data augmentation technique which alters the speed of an acoustic signal. It can either speed up or slow down the acoustic signal and the spectral content will remain unchanged. Given the input acoustic signal, $\mathbf{x}$, time stretching stretches $\mathbf{x}$ by dividing it into overlapping blocks with interval R_a, where R denotes the interval between the blocks, and a indicates the initial value. Subsequently, it is reassembled with different spacing R_s, where s denotes the target value. The output time stretched augmented signal has a rate of $\alpha = \frac{R_a}{R_s}$ with respect to the initial input signal. Although this approach fulfills the fundamental requirements of time stretching, the time stretched signals are phase discontinuities due to the re-spacing of the acoustic blocks. In view of this, phase vocoder algorithm is applied to overcome the phase discontinuities issue. By using short-time Fourier Transform (STFT), $\mathbf{x}$ is partitioned into collective of overlapping frames, $\overline{\mathbf{X}}_{frame}$. The amplitude and phase information for every frequency channels at specific time instants, $t_a^u = u \cdot R_a (u \in U)$ of the input signal are then obtained where U denotes the total number of frames. Discrete Fourier transform (DFT) is computed to transform the frame into frequency domain and obtain the successive spectral representations:

$$X\left(t_a^u, k\right) = \sum_{j=1}^{J} \mathbf{x}_u \left(t_a^u + j\right) Hann(j) \cdot e^{-\varphi \Omega_k j}, \quad \Omega_k = \frac{2\pi k}{J} \quad (1)$$

where $1 \leq k \leq J$ and φ is the imaginary unit, where $\varphi = \sqrt{(-1)}$. In addition, the Hanning window function, $Hann(j)$, is used in both time stretching and pitch shifting techniques. Generally, $\mathbf{x}_u$ possesses the quasi-stationary characteristic. Therefore, phase adjustment can be done via altering the phase of each bin to match the phases at shifted synthesis time positions, t_a^u. In addition, phase adjustment depends on the observed phase difference between the previous and current frame. Hence, spectrum is divided into amplitude, $r\left(t_a^u, k\right)$, and phase, $\Phi\left(t_a^u, k\right)$:

$$\begin{gathered} r\left(t_a^u, k\right) = \left|X\left(t_a^u, k\right)\right| \\ \Phi\left(t_a^u, k\right) = \arg\left(X\left(t_a^u, k\right)\right) \end{gathered} \quad (2)$$

The phase increment for the synthesis hop size can be obtained with the multiplication between $\omega\left(t_a^u, k\right)$ and R_s. Given the phase increment for synthesis hop size, $R_s\omega\left(t_a^u, k\right)$, the output phase can be calculated by adding the phase from the previous frame:

$$\Psi\left(t_s^u, k\right) = \Psi\left(t_s^{u-1}, k\right) + R_s \omega\left(t_a^u, k\right) \quad (3)$$

With the output phase obtained from (3), synthesis frames y_u at t_a^u are computed by applying an inverse fourier transformation:

$$\begin{gathered} Y\left(t_s^u, k\right) = r\left(t_a^u, k\right) \cdot e^{-\varphi \Psi\left(t_s^u, k\right)} \\ y_u(j) = \frac{1}{J} \sum_{k=1}^{K} Y\left(t_s^u, k\right) \cdot e^{\varphi \Omega_k j} \end{gathered} \quad (4)$$

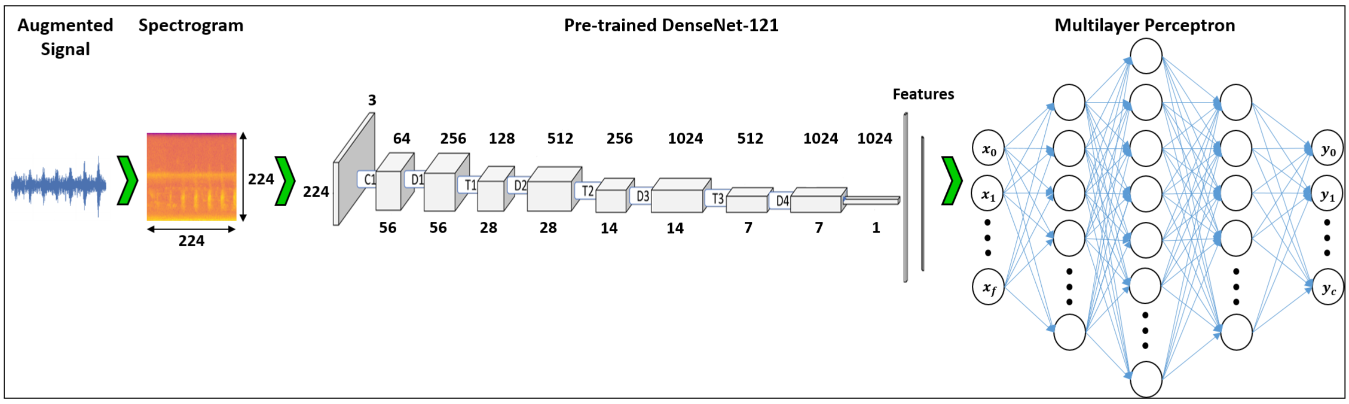

Fig. 1. Architecture of the Proposed Pre-trained DenseNet-121 with Multilayer Perceptron

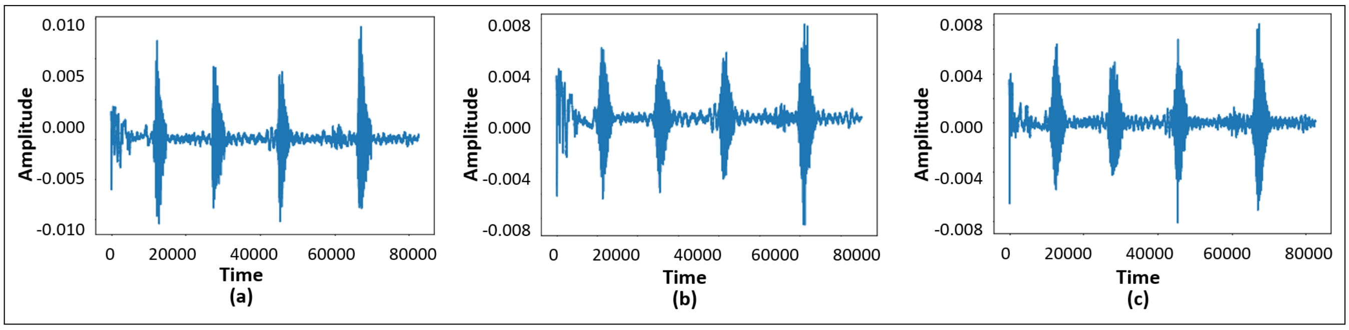

Fig. 2. Data Augmentation: (a) Original Signal, (b) Time Stretching Signal, (c) Pitch Shifting Signal

By applying windowing and overlap-adding of the single frames, the final time-stretched augmented signal can be obtained via:

$$\mathbf{x}_{stretch} = \sum_{u=1}^{U} Hann\left(j - t_s^u\right) \cdot y_u\left(j - t_s^u\right) \tag{5}$$

2) Pitch Shifting: Dissimilar to the time stretching approach, pitch shifting technique attempts to alter the original pitch of the audio data. Pitch shifting increases or decreases the pitch of the acoustic data. In this work, pitch shifting is applied via phase vocoder to produce pitch shifted acoustic signals. Phase vocoder is commonly used to stretch the signal in frequency domain which solves phasiness by utilising STFT. To apply pitch shifting to the acoustic signal, all the frequency components in an audio signal are multiplied by a pitch shift ratio without affecting the signal duration. First, the acoustic signal, $\mathbf{x}$, is partitioned into several blocks via STFT. After that, it undergoes conversion from time-domain into frequency-domain. Frequency bins are computed by employing DFT which transforms the block into frequency domain. Equation (2) is used to calculate the amplitudes and phases as the estimation for the time point at the block centre. Moreover, the phase increment between two analysis points is computed and all the frequency bins computed are then scaled with the principal argument presented in [26]. By adding the phase of the previous frame, the final phase can be computed with (3). Both altered phases $\Phi\left(t_a^u, k\right)$ and amplitudes $r\left(t_a^u, k\right)$ are then transformed back into blocks of samples using inverse fourier transformation and Hanning window function, $Hann(j)$:

$$Hann(j) = 0.5 - 0.5\cos\left(\frac{2\pi j}{J}\right), \ 1 \leq j \leq J \tag{6}$$

The output time stretched signal then undergoes a resampling process. Given α denotes the stretch rate, S_{initial} denotes the initial sampling rate, the target sampling rate, S_{target} is computed as:

$$S_{\text{target}} = \frac{S_{\text{initial}}}{\alpha} \tag{7}$$

Two signal processing techniques: trimming and padding, are applied to the resulting resampled acoustic signal, $\mathbf{x}_{resample}$, to ensure the output length is equivalent to the input length. Trimming technique attempts to trim the output resampled acoustic signal, $\mathbf{x}_{resample}$, to a target length when the length of the resampled acoustic signal, $\mathbf{x}_{resample}$, is larger than the length of input signal. On the other hand, the padding process will pad the resampled acoustic signal, $\mathbf{x}_{resample}$, with trailing zeros when the length of the resampled acoustic signal, $\mathbf{x}_{resample}$, is smaller than the length of input signal. Based on the length of the output resampled acoustic signal $\mathbf{x}_{resample}$, a pitch shifted acoustic signal, $\mathbf{x}_{pitch}$, is obtained by applying either trimming or padding.

B. Frequency Spectrogram

After data augmentation is applied to the training set, the augmented training samples are converted into frequency spectrograms before it is fed into the pre-trained DenseNet-121 model. In general, an acoustic signal can be visualised in time-domain representation in which the loudness of the acoustic signal over the time is presented. The amplitude information in this conventional visualisation method is not very informative because it only indicates the loudness of the acoustic signal. This is where frequency spectrograms come into the play. Frequency spectrogram is a popular method to visualise an audio signal. It is able to represent the time, frequency, and amplitude information in a graph. Generally, frequency spectrogram is a two-dimensional graph with an additional third dimension that is represented by colours. It shows the signal strength over the time at various frequencies. The amplitude of a specific frequency at a particular time is visualised using colour.

The idea of frequency spectrogram technique is to divide the acoustic signal into small windows. The Hanning window in (6) is applied to smooth the discontinuities of small signals. Subsequently, DFT is computed for each small window. In this work, windows overlapping technique is adopted to minimise the loss of frequencies. For a single small window of the input acoustic signal, DFT converts it into frequency space using:

$$X\left(k\right) = \sum_{j=1}^{J} \mathbf{x}_{aug_o}\left(j\right) \cdot e^{-\varphi \Omega_k j}, \quad \Omega_k = \frac{2\pi k}{J} \tag{8}$$

where $\mathbf{x}_{aug_o}$ represents the augmented acoustic signal, $o = 1, 2, \ldots, O$ denotes index of the augmented acoustic signal, and j denotes the index of the windows in $\mathbf{x}_{aug_o}$. The index of the windows in X is denoted by k, where $k = 1, 2, \ldots, K$. The imaginary unit φ, where $\varphi = \sqrt{(-1)}$, converts the real sinusoids into complex sinusoids. The output $X(k)$ is a sequence of coefficients with the length of K. The same DFT is then applied to all windows and the time is represented by the sequential of the windows. The output coefficients represent the amplitudes of the different frequencies in each of the small windows. Subsequently, the resulting output, $X(k)$, is normalised. In this work, min-max feature scaling method is used to perform normalisation that brings all values into the range $[0, 1]$:

$$X(k)' = \frac{X(k) - X(k)_{\min}}{X(k)_{\max} - X(k)_{\min}} \tag{9}$$

where $X(k)'$ denotes the normalised value of $X(k)$. The resulting spectrums form a three-dimensional frequency spectrogram image, $\mathbf{x}_{spec_o}$, where the rows and columns represent the number of window frames and the frequency bin, respectively. Additionally, the colours of the frequency spectrogram image represent the strength of the frequencies. As compared to 1D acoustic signal that only contains temporal information, spectrogram image provides more information of the audio signal that includes time, frequency, and amplitude of the acoustic signal. The dimension of the spectrogram image is denoted as $(h \times w) \times d$, which represents rows, columns, and channels, respectively. The output frequency spectrogram images, $\mathbf{x}_{spec_o}$, will be passed to the next phase to extract the significant features.

C. Transfer Learning with Pre-trained DenseNet-121

Transfer learning is a machine learning approach that attempts to use an architecture that is already learned from another domain, and apply that knowledge to other target tasks instead of starting from scratch. In the deep learning field, transfer learning is a popular approach where pre-trained models that are trained with large-scale data are used to accelerate the training process and improve the performance of the deep learning model. With the transfer learning approach, the parameters from the pre-trained model are transferred to a new task. Due to the limited training samples in the dataset, transfer learning approach is particularly useful which provides good initialisation for the parameters. Although the datasets used in this work are acoustic data, it is converted into a spectrogram image with the frequency spectrogram technique that is presented in Section III-B. Therefore, it is more appropriate to transfer a pre-trained model from an image-related task. The knowledge of the pre-trained model is migrated in this research to extract features from the frequency spectrogram image. In this work, a pre-trained DenseNet-121 model on ImageNet is proposed to extract robust features from each frequency spectrogram image, $\mathbf{x}_{spec_o}$. The softmax layer of the pre-trained DenseNet-121 is removed because the classification process is not needed in this stage.

Densely Connected Convolutional Networks (DenseNet) is introduced to encounter the gradient vanishing problem in Convolutional Neural Networks (CNN) architecture. DenseNet is a CNN architecture that utilises dense connections among all the layers that simplifies the connectivity pattern between layers yet ensures maximum information flow. DenseNet encourages feature reuse that include the feature maps from other layers as the additional inputs. The feature maps of a layer, $\mathbf{z}$, are used as additional inputs for other subsequent layers. This connectivity pattern is known as dense connectivity. DenseNet attempts to concatenate the resulting feature maps to impede information flow. Hence, feature map of ℓ^{th} layers in DenseNet receives feature maps from all preceding layers, $\mathbf{z}_0, \mathbf{z}_1, \ldots, \mathbf{z}_{\ell-1}$, as the inputs:

$$\mathbf{z}_\ell = H_\ell\left(\left[\mathbf{z}_0, \mathbf{z}_1, \ldots, \mathbf{z}_{\ell-1}\right]\right) \tag{10}$$

where $[\mathbf{z}_0, \mathbf{z}_1, \ldots, \mathbf{z}_{\ell-1}]$ refers to concatenated feature maps from layers $0, \ldots, \ell-1$. $H_\ell(\cdot)$ is a composite function that is inspired by [27]. The composite function comprises of three consecutive operations, which are: (1) batch normalisation (BN) [28], (2) interleaved with a rectified linear unit (ReLU) [29], and followed by (3) a 3×3 convolution (Conv). A problem arises in dense connectivity is that the feature maps with different sizes are not viable. In view of this, DenseNet is divided into several densely connected blocks, known as DenseBlocks. Each DenseBlocks has a constant dimension of the feature maps regardless of the number of filters. It helps to facilitate down sampling in DenseNet to reduce the size of feature maps. The layer between two DenseBlocks is referred to as the transition layer. Each transition layer comprises a batch normalisation layer, a 1×1 convolutional layer, and followed by a 2×2 average pooling layer.

Channel dimension of DenseNet will be increased because the feature maps are concatenated at each layer. Let each H_ℓ produce c feature maps where this hyperparameter c is refers to the growth rate of the network that regulates the amount of information added to the network in each layer. c_ℓ can be generalised as below:

$$c_\ell = c_0 + c \times (\ell - 1) \tag{11}$$

where the number of channels for the input layer is denoted by c_0. In contrast to other deep neural network architectures, DenseNet can have very narrow layers. Every layer has a capability to access to its preceding feature maps as the network's collective knowledge. At each layer, new information, i.e., c feature maps of information, is added to this collective knowledge. Moreover, an average pooling is utilised to compute the average value for the patches of a feature map. One of the advantages of DenseNet-121 is strong gradient flow where the error signals are back-propagated to earlier layers directly due to the dense connectivity. This dense connection allows the earlier layers to get direct supervision from the output layer that is known as implicit deep supervision. Moreover, DenseNet-121 practises a dense connectivity pattern, and thereby, the inputs for each layer is the network's collective knowledge containing diversified features. This has produced richer patterns as compared to other deep neural network architectures. The output flatten feature vector, $\mathbf{f}_o$, of pre-trained DenseNet-121 is then passed to the multilayer perceptron for acoustic event classification.

D. Multilayer Perceptron

Multilayer perceptron (MLP) is a feedforward artificial neural network comprising at least three layers of neurons, which are an input layer, a hidden layer, and an output layer. In this paper, the features derived from the pre-trained DenseNet-121, $\mathbf{f}_o$ are fed into the MLP as the input data. Therefore, the number of neurons in the input layer is equal to the size of the extracted features. For the hidden layer, the number of neurons is decided empirically in order to optimise the network with better generalisation for AEC.

MLP is a supervised learning algorithm which performs training on a set of input-output pairs. MLP models the correlation of the input-output pairs. Learning of MLP is an operation of updating the weight connections in order to minimise the difference between predicted output and the ground truth. Learning takes place in every perceptron of the MLP model where the weight connections are updated after each chunk of data has been processed. The modification of the weight connections is based on the error rate of the difference between the predicted output and the ground truth. During the training process for a MLP, the parameters: weights and biases, are adjusted in order to minimise the error rate. A forward pass and a backward pass are performed in the training. In the forward pass, the input data is propagated from the input layer to the hidden layers, and accordingly reaches the output layer. On the other hand, backward pass attempts to back propagate the gradient weights and biases by utilising backpropagation. Backpropagation is an approach to generalise the least mean squares algorithm for supervised learning of artificial neural networks (ANN) using gradient descent. Given $\mathbf{f}_o$ as the input data with a feature size of Q, and thereby, an input layer with Q neurons, a ReLU activation function is used in the proposed MLP:

$$g(x) = \max(0, x) \tag{12}$$

ReLU outputs a zero for negative input and returns its argument x for positive inputs. The outputs of the neurons from the hidden layer, C^l can be computed by squash the total net input, net^l with the ReLU activation function, $g(x)$:

$$net^l = \begin{cases} \sum_{i=1}^{I^l} w_i^l \cdot \mathbf{f}_{o_i} + \theta^l, & \text{if } l = 1 \\ \sum_{i=1}^{I^l} w_i^l \cdot C_i^{l-1} + \theta^l, & l = 2, 3, \ldots, L \end{cases} \tag{13}$$

$$C^l = g\left(net^l\right) \tag{14}$$

where I^l is the amount of inputs for the l^{th} hidden layer neuron, θ^l denotes the bias of l^{th} layer, and w_i^l denotes the weights of l^{th} layer with i^{th} neuron. When $l = 1$, the weights and input data are used to compute the total net input of the l^{th} hidden layer neuron. To calculate the net input of the l^{th} layer where $l = 2, 3, \ldots, L$, the output of the $(l-1)^{th}$ layer, C_i^{l-1}, is taken as the inputs; and the weights, w_i^l, are included as presented in 13. Thereafter, ReLU activation function, $g(x)$ is applied to the total net input, net^l to compute the neuron's output, C^l. After the outputs of the neurons in output layer is computed, the degree of error in an output neuron is obtained as below:

$$E_i = G_i^L - C_i^L,\ i = 1, 2, \ldots, I \tag{15}$$

where G_i^L and C_i^L denote the target output and the computed output of the network, respectively. Given the error rate of each output neuron, E_i, the total error of the network can be calculated as:

$$E_{total} = \sum_{i=1}^{I^L} \frac{1}{2} E_i^2 \tag{16}$$

Subsequently, in order to reduce the total error, each weight, w_i^l, in the network is updated using backpropagation. The partial derivative of E_{total} with respect to w_i^l determines how much change in w_i^l affects the total error E_{total} and it can be computed by applying chain rule:

$$\frac{\partial E_{total}}{\partial w_i^l} = \frac{\partial E_{total}}{\partial C_i^l} * \frac{\partial C_i^l}{\partial net_i^l} * \frac{\partial net_i^l}{\partial w_i^l} \tag{17}$$

where total error of the network, E_{total}, change with respect to the output neuron, C_i^l, is denoted as:

$$\frac{\partial E_{total}}{\partial C_i^l} = \begin{cases} -\left(E_i\right), & \text{if } l = L \\ \sum_{i=1}^{I^l} \frac{\partial \frac{1}{2} E_i^2}{\partial C_i^l}, & l = 1, 2, \ldots, L-1 \end{cases} \tag{18}$$

For $l = L$, the total error change with respect to the output neurons can be evaluated as $-\left(E_i\right)$. For $l = 1, 2, \ldots, L-1$, the output of each hidden layer neuron contributes to the output of multiple output layer neurons. It is notable that there are i number of output neurons in the l^{th} layer. Therefore, the total error change with respect to the output neurons can be rewritten as, $\sum_{i=1}^{I^l} \frac{\partial \frac{1}{2} E_i^2}{\partial C_i^l}$, as presented in (18). Additionally, each $\frac{\partial \frac{1}{2} E_i^2}{\partial C_i^l}$ can be further computed as:

$$\frac{\partial \frac{1}{2} E_i^2}{\partial C_i^l} = \frac{\partial \frac{1}{2} E_i^2}{\partial net_i^l} * \frac{\partial net_i^l}{\partial C_i^l} \tag{19}$$

Given the derivative of the ReLU activation function, $g'(x)$,

$$g'(x) = \left\{ \begin{array}{lll} 0 & \text{for} & x < 0 \\ 1 & \text{for} & x \geq 0 \end{array} \right\} \tag{20}$$

the output of C_i^l change with respect to its total net input, net_i^l, in (17) is computed as:

$$\frac{\partial C_i^l}{\partial net_i^l} = g'(C_i^l) \tag{21}$$

and the total net input of neurons, net_i^l, change with respect to the weight, w_i^l is calculated as:

$$\frac{\partial net_i^l}{\partial w_i^l} = C_i^{l-1} \tag{22}$$

With the partial derivative of E_{total} with respect to w_i^l, the weight is updated by subtracting $\frac{\partial E_{total}}{\partial w_i^l}$ from the current weight:

$$\nabla w_i^l = w_i^l - \eta \frac{\partial E_{total}}{\partial w_i^l} \tag{23}$$

where η denotes the learning rate. This hyperparameter is used to control how fast the parameters converge. This training process is repeated to update the weights until the criterion for termination is achieved. Thereafter, the output of the neurons in the output layer are interpreted as a probability of the event classes by applying softmax function:

$$\sigma(C^L)_v = \frac{e^{C^L{}_v}}{\sum_{v=1}^{V} e^{C^L{}_v}}, \; v = 1, 2, \ldots, V \tag{24}$$

where the total number of classes is denoted by V. C^L denotes the output from the output layer and it is used as the input into the softmax function. The denominator $\sum_{v=1}^{V} e^{C^L{}_v}$ is used to ensure the summation of all output is equivalent to 1 in which to provide a probability distribution. The predicted probability distribution is computed as:

$$\hat{\mathbf{y}} = \begin{pmatrix} \sigma(C^L)_1 \\ \sigma(C^L)_2 \\ \vdots \\ \sigma(C^L)_V \end{pmatrix} \tag{25}$$

where $\hat{\mathbf{y}}$ denotes the predicted probability for each event classes and the final predicted class is based on the maximum probability:

$$\rho_{MLP} = \arg\max(\hat{\mathbf{y}}) \tag{26}$$

The training of the proposed pre-trained DenseNet-121 with MLP is presented in Algorithm 1.

Algorithm 1 Training in Pre-trained DenseNet-121 with MLP.

```
 1: Initialisation: All parameters and hyperparameters
 2: for epoch=1 to epoch=MaximumEpoch do
 3:     for iter=1 to iter=MaximumIteration do
 4:         $Input$ ← Pre-trained DenseNet-121 features.
 5:         for $o$=1 to $o$=$O$ do
 6:             Forward propagation: $net^l$ (refer to (13))
 7:             Backward propagation: $\frac{\partial E_{\text{total}}}{\partial C_i^l}$ (refer to (18))
 8:         end for
 9:         Output layer calculation: $\sigma(C^L)_v$ (refer to (24))
10:         Gradient calculation: $\frac{\partial E_{\text{total}}}{\partial w_i^l}$ (refer to (17))
11:         Parameters update: $\nabla w_i^l$ (refer to (23))
12:     end for
13: end for
```

IV. EXPERIMENTS

This section describes the experimental details, including dataset, ablation study, and experimental results in comparison with existing methods.

A. Datasets

In this work, three acoustic event datasets, namely Soundscapes1 (Sound1), Soundscapes2 (Sound2), and UrbanSound8K (Urban) datasets are used to evaluate the performance of the proposed pre-trained DenseNet-121 with MLP. Soundscapes1 and Soundscapes2 datasets are collected by Motorola Solutions Malaysia Sdn. Bhd. Soundscapes1 dataset is a small-scale dataset that contains 388 acoustic event samples. The dataset contains eight different acoustic event classes: *birds, canyon, laugh, murmur, thunder, war, water, and wind*. The number of samples for each event class varies. Soundscapes2 dataset consists of 235 acoustic event samples that are grouped into ten event classes: *assembly, conveyor, cutter, elevators, grinding, hammer, motor combustion, printer, relay, and welding*. Another database, namely, UrbanSound8K dataset [30] contains a total number of 8,732 labelled acoustic samples categorised into ten different acoustic event classes: *air conditioner, car horn, children playing, dog bark, drilling, engine idling, gun shot, jackhammer, siren, and street music*. In this work, each dataset is split into 80% training samples and 20% testing samples. Two data augmentation techniques: time stretching and pitch shifting are applied to the train set to boost the number of training samples. All augmented acoustic samples are undersampled into 22,050 kHz sampling rate which is between the lower and upper limits of human hearing.

B. Ablation Study

The ablation analysis of the pitch shifting, time stretching, and frequency spectrogram image with Pre-trained DenseNet-121 with MLP model on the three datasets are presented in TABLE I. The experiment is first carried out without any data augmentation and data pre-processing techniques and the result is used as the baseline performance of the MLP model. Thereafter, time stretching technique is applied and the F1-score is enhanced by 3.7%, 11.6%, and 5.3% on Soundscapes1, Soundscapes2, and UrbanSound8K datasets, respectively. Later, another data augmentation technique, pitch shifting is implemented to boost the number of training samples. By employing both data augmentation techniques, the performance of the proposed model is further improved by 7.4%, 13.6%, and 8.1% for Soundscapes1, Soundscapes2, and UrbanSound8K datasets, respectively. This has proved that by implementing both of the data augmentation techniques, more training samples are obtained and this has enabled the model to better encode the representative features. Therefore, the model has improved the generalisation ability and it is less prone to overfitting. Additionally, the pre-trained DenseNet-121 model is used to extract compact features from the frequency spectrogram image. Thereafter, extracted significant features are fed into

MLP for training which further improve the performance of the model with F1-score of 80.7%, 87.3%, and 69.6% on Soundscapes1, Soundscapes2, and UrbanSound8K datasets, respectively.

TABLE I
THE ABLATIVE ANALYSIS OF PRE-TRAINED DENSENET-121 WITH MLP ON SOUNDSCAPES1, SOUNDSCAPES2, AND URBANSOUND8K DATASETS, MEASURED IN F1-SCORE (%)

Methods	**Sound1**	**Sound2**	**Urban**
MLP	63.0	49.3	45.5
MLP + Time Stretching	66.7	60.9	50.8
MLP + Time Stretching + Pitch Shifting	74.1	74.5	58.9
MLP + Time Stretching + Pitch Shifting + Pre-trained DenseNet-121 features	**80.7**	**87.3**	**69.6**

In neural networks-based research, hyperparameters setting of the proposed methods is important in order to achieve good performance. In view of this, the optimal values of numerous hyperparameters of the proposed Pre-trained DenseNet-121 with MLP is determined through the experiments. The hyperparameters included in this work are the No. of hidden layers, No. of hidden states, batch size, and learning rate. As for the Pre-trained DenseNet-121 hyperparameters, exactly the same hyperparameters settings reported in [31] are used in this work. The number of hidden layers is initialised to one layer, and incrementally adds one layer, until no improvement in performance is observed. The number of hidden states is determined in the set of $\{64, 128, 256, 512\}$ which is used throughout all hidden layers. For the batch size, the initial value is set to 16, and increments by 16 until it reaches a maximum number of 64. The maximum number of training epochs is set to 100 in all experiments. On the other hand, the values of the learning rate are determined in the set of $\{0.001, 0.0001\}$ to ensure the weights converge steadily. The summary of the optimal hyperparameters settings of the proposed method on three datasets is shown in TABLE II.

TABLE II
SUMMARY OF THE OPTIMAL HYPERPARAMETERS FOR THREE DATASETS

Hyper-parameters	**No. of hidden layers**	**No. of hidden states**		**Batch size**	**Learn-ing rate**
Tested value	1, 2, 3, 4	64, 128, 256, 512		16, 32, 64	0.001, 0.0001
Optimal Hyperparameters Settings					
Dataset	**No. of hidden layers**	**No. of hidden states**	**Batch size**	**Learn-ing rate**	**F1-score (Epoch)**
Sound1	3	(64, 512, 64)	64	0.001	80.7 (87)
Sound2	4	(512, 256, 128, 256)	16	0.0001	87.3 (96)
Urban8K	4	(128, 256, 512, 512)	16	0.0001	69.6 (78)

C. Comparison with the Existing Methods

The performance of the proposed pre-trained DenseNet-121 with MLP is compared with the state-of-the-art acoustic event models in terms of F1-score to further evaluate the performance of the proposed solution in AEC tasks. The existing acoustic event models under consideration are mainly based on neural networks, which include Pre-trained VGG with CNN [32], SB-CNN [21], and NMF-CNN [33]. These models are assessed on Soundscapes1, Soundscapes2, and UrbanSound8K acoustic event datasets, and the experimental results are listed in TABLE III.

As presented in TABLE III, the proposed method outperforms all three existing methods on Soundscapes1, Soundscapes2, and UrbanSound8K datasets. As shows in the table, the proposed Pre-trained DenseNet-121 with MLP outperforms the Pre-trained VGG with CNN method [32] in Soundscapes1, Soundscapes2 and UrbanSound8K datasets, with a difference in the F1-score of 1.9%, 1.4%, and 0.6%, respectively. The proposed model also outshines the SB-CNN method [21] on the Soundscapes1, Soundscapes2 and UrbanSound8K datasets, with a difference in F1-score of 4.2% , 18.2%, and 0.6%, respectively. Likewise, the proposed model performs better than the NMF-CNN [33] with a difference of 7%, 8.4%, and 3.2% in F1-score, on the Soundscapes1, Soundscapes2 and UrbanSound8K datasets, respectively.

Two data augmentation techniques utilised in this work boost the training samples and help in reducing the overfitting of the model. Besides, more information of the signal can be obtained by converting the acoustic signal into frequency spectrogram image which is a large contribution to improve the performance of the proposed method. Through transfer learning with a pre-trained DenseNet-121 model able to accelerate the training process and extract significant features from the spectrogram which greatly improve the performance of the proposed method. Lastly, the performance of the proposed method is further improved due to the capability of MLP to learn complex relationships in mapping the input to the output of the training samples which is important in addressing non-linear and complex problems.

TABLE III
COMPARISON IN F1-SCORE FOR PROPOSED METHODS WITH OTHER EXISTING METHODS

Methods	**Sound1**	**Sound2**	**Urban**
Pre-trained VGG with CNN [32]	78.8	85.9	65.3
SB-CNN [21]	76.5	69.1	69.0
NMF-CNN [33]	73.7	78.9	66.4
Pre-trained DenseNet-121 with MLP	**80.7**	**87.3**	**69.6**

D. t-SNE Visualisation

To project high-dimensional hidden states into a lower dimensional, t-SNE [34] approach is implemented to project the hidden states of the acoustic event samples into the 2D space. All the acoustic event classes from the three benchmark datasets are analysed and visualised using t-SNE visualisation technique.

The t-SNE visualisation of the proposed methods on the Soundscapes1, Soundscapes2, and UrbanSound8K dataset are depicted in Fig. 3, Fig. 4, and Fig. 5, respectively. Noticeably, several distinct clusters are formed for acoustic events without mixing with other classes and they are linearly separable. Conversely, some clusters overlap with other classes of the acoustic event, where their decision boundaries are not linearly separable. Specifically, *Canyon* and *Wind* in Soundscapes1 dataset which is likely to lead to classification error. For the Soundscapes2 dataset, most of the acoustic event classes like *Assembly, Hammer, Relay*, and *Cutter* are linearly separable with Pretrained DenseNet-121 with MLP.

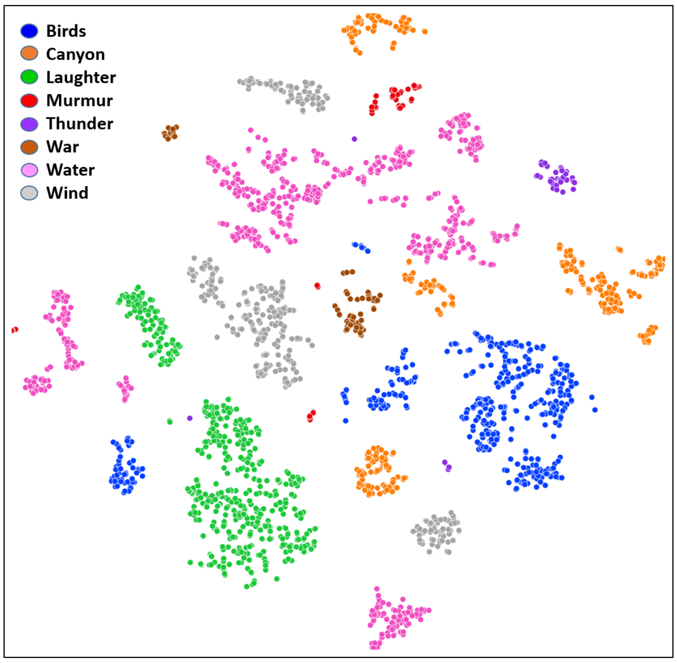

Fig. 3. t-SNE of the Proposed Method on Soundscapes1 Dataset

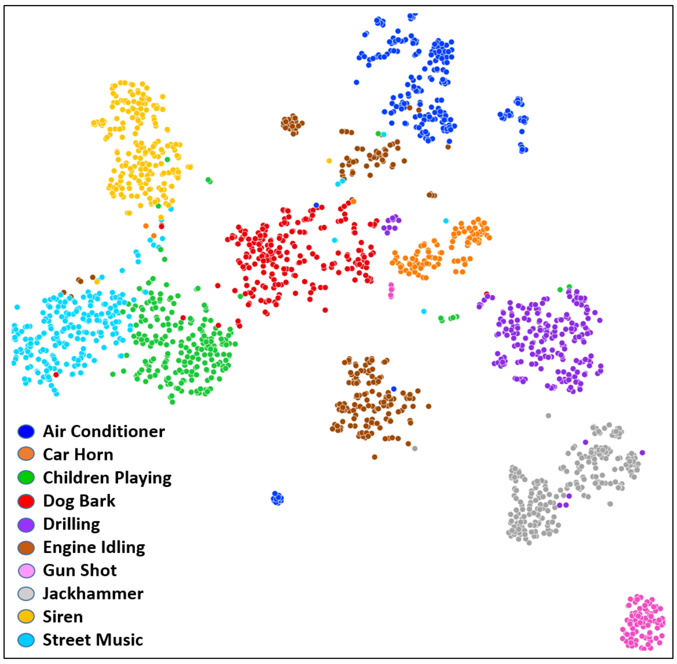

Fig. 5. t-SNE of the Proposed Method on UrbanSound8K Dataset

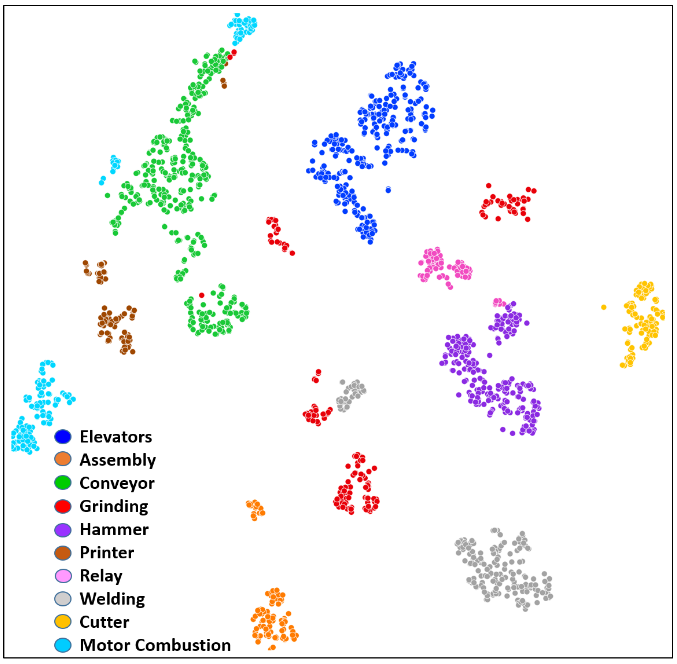

Fig. 4. t-SNE of the Proposed Method on Soundscapes2 Dataset

However, some classes like *Welding*, *Grinding* and *Conveyor* still lack a good representation to form the clusters well. As presented in the Fig. 5, several acoustic events classes from the UrbanSound8K dataset mix with other classes and they are non-linearly separable. Specifically, *Children Playing* versus *Dog Bark*, *Children Playing* versus *Street Music*, and *Air Conditioner* versus *Engine Idling*. It is worth noting that the samples of *Air Conditioner* and *Drilling* form more distinct clusters with the proposed Pre-trained DenseNet-121 with MLP. Concluding from the above analysis, the proposed Pre-trained DenseNet-121 with MLP performs well in AEC tasks.

E. *Confusion Matrix Visualisation*

To further evaluate the performance of the proposed method, a confusion matrix visualisation technique is utilised. All the acoustic events from three benchmark datasets are analysed. Confusion matrices for proposed Pre-trained DenseNet-121 with MLP on Soundscapes1, Soundscapes2, and UrbanSound8K are demonstrated in Fig. 6, Fig. 7, and Fig. 8, respectively.

In the Fig. 6, two acoustic event classes from Soundscapes1 dataset are 100% correctly classified, which are War and Wind. It is notable that Canyon class is heavily misclassified to Water class by 66.67% of the test samples due to the recording background for Canyon consists of river flow noise. For Soundscapes2 dataset, a total of six out of ten event classes are 100% correctly classified. In addition, most of the leftover event classes are well classified with Pre-trained DenseNet-121 with MLP which achieved 88.24%, 80%, and 80% for Elevators, Conveyor, and Printer, respectively.

Fig. 8 presents the confusion matrix of Pre-trained DenseNet-121 with MLP for UrbanSound8K dataset. Based on the resulting confusion matrix depicted in Fig. 8, Dog Bark, Engine Idling, Gun Shot, Jackhammer, and Children Playing are well classified with the proposed method which achieved 84.15%, 82.26%, 81.4%, 76.03%, and 75.23%, respectively. The confusion matrices show that the proposed pre-trained DenseNet-121 with MLP able to obtain robust features from the acoustic samples and classify them into correct classes with promising performance.

V. Conclusion

In this paper, a pre-trained DenseNet-121 with MLP method is proposed for acoustic event classification. Two data augmentation techniques, namely, time stretching and pitch shifting are applied on the training set to expand the number of training samples. Additionally, frequency

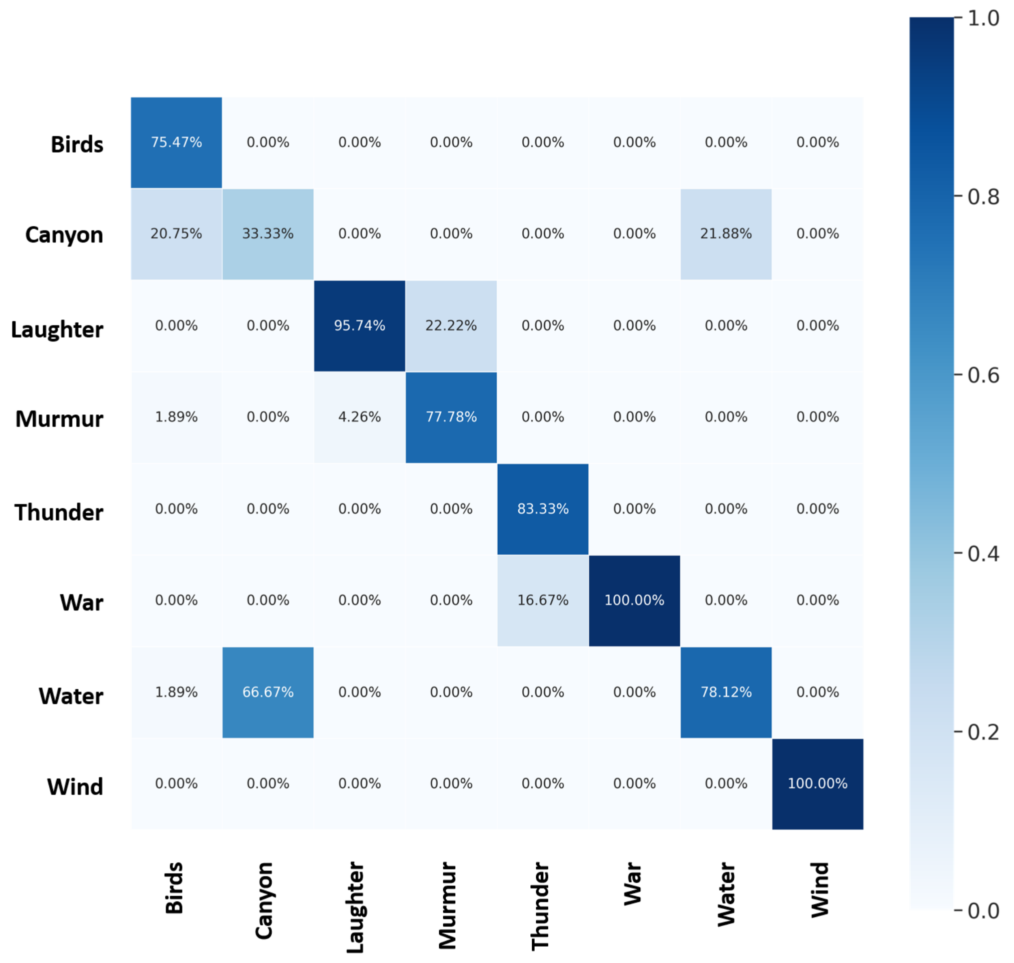

Fig. 6. Confusion Matrix of the Proposed Method on Soundscapes1 Dataset

spectrogram technique is used to convert acoustic signals into frequency spectrogram images which contain more information and simultaneously solve the varying length problem of the acoustic signal. Moreover, a set of significant features can be extracted from the spectrogram images by implementing transfer learning with the pre-trained DenseNet-121 model. The extracted significant features vector is then fed into a MLP model for training and classification. Three acoustic event datasets: Soundscapes1, Soundscapes2, and UrbanSound8K are used to evaluate the performance of the proposed pre-trained DenseNet-121 with MLP method. The proposed method outshines the existing methods in comparison and yields the F1-score of 80.7%, 87.3%, and 69.6% on Soundscapes1, Soundscapes2, and UrbanSound8K datasets, respectively.

REFERENCES

[1] C. P. Lee, K. M. Lim, T. J. Yu, and S. F. Abdul Razak, "Ai-based targeted advertising system," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 13, no. 2, pp. 787–793, 2019.

[2] J. F. Gemmeke, L. Vuegen, P. Karsmakers, B. Vanrumste *et al.*, "An exemplar-based nmf approach to audio event detection," in *2013 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*. IEEE, 2013, pp. 1–4.

[3] J. Ludeña-Choez and A. Gallardo-Antolín, "Nmf-based temporal feature integration for acoustic event classification." pp. 2924–2928, 2013.

[4] L. Vuegen, B. Broeck, P. Karsmakers, J. F. Gemmeke, B. Vanrumste, and H. Hamme, "An mfcc-gmm approach for event detection and classification," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2013, pp. 1–3.

[5] A. Temko, R. Malkin, C. Zieger, D. Macho, C. Nadeu, and M. Omologo, "Acoustic event detection and classification in smart-room environments: Evaluation of chil project systems," *Cough*, vol. 65, no. 48, p. 5, 2006.

[6] W. Nogueira, G. Roma, and P. Herrera, "Automatic event classification using front end single channel noise reduction, mfcc features and a support vector machine classifier," *IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events*, pp. 1–2, 2013.

[7] X. Zhuang, X. Zhou, M. A. Hasegawa-Johnson, and T. S. Huang, "Real-world acoustic event detection," *Pattern Recognition Letters*, vol. 31, no. 12, pp. 1543–1551, 2010.

[8] J. Schröder, N. Moritz, M. R. Schädler, B. Cauchi, K. Adiloglu, J. Anemüller, S. Doclo, B. Kollmeier, and S. Goetze, "On the use of spectro-temporal features for the ieee aasp challenge 'detection and classification of acoustic scenes and events'," in *2013 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*. IEEE, 2013, pp. 1–4.

[9] Y. Ephraim and D. Malah, "Speech enhancement using a minimum-mean square error short-time spectral amplitude estimator," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 32, no. 6, pp. 1109–1121, 1984.

[10] R. Martin, "Noise power spectral density estimation based on optimal smoothing and minimum statistics," *IEEE Transactions on Speech and Audio Processing*, vol. 9, no. 5, pp. 504–512, 2001.

[11] X. Xia, R. Togneri, F. Sohel, and D. Huang, "Random forest classification based acoustic event detection utilizing contextual-information and bottleneck features," *Pattern Recognition*, vol. 81, pp. 1–13, 2018.

[12] H. Phan, M. Maaß, R. Mazur, and A. Mertins, "Random regression forests for acoustic event detection and classification," *IEEE/ACM*

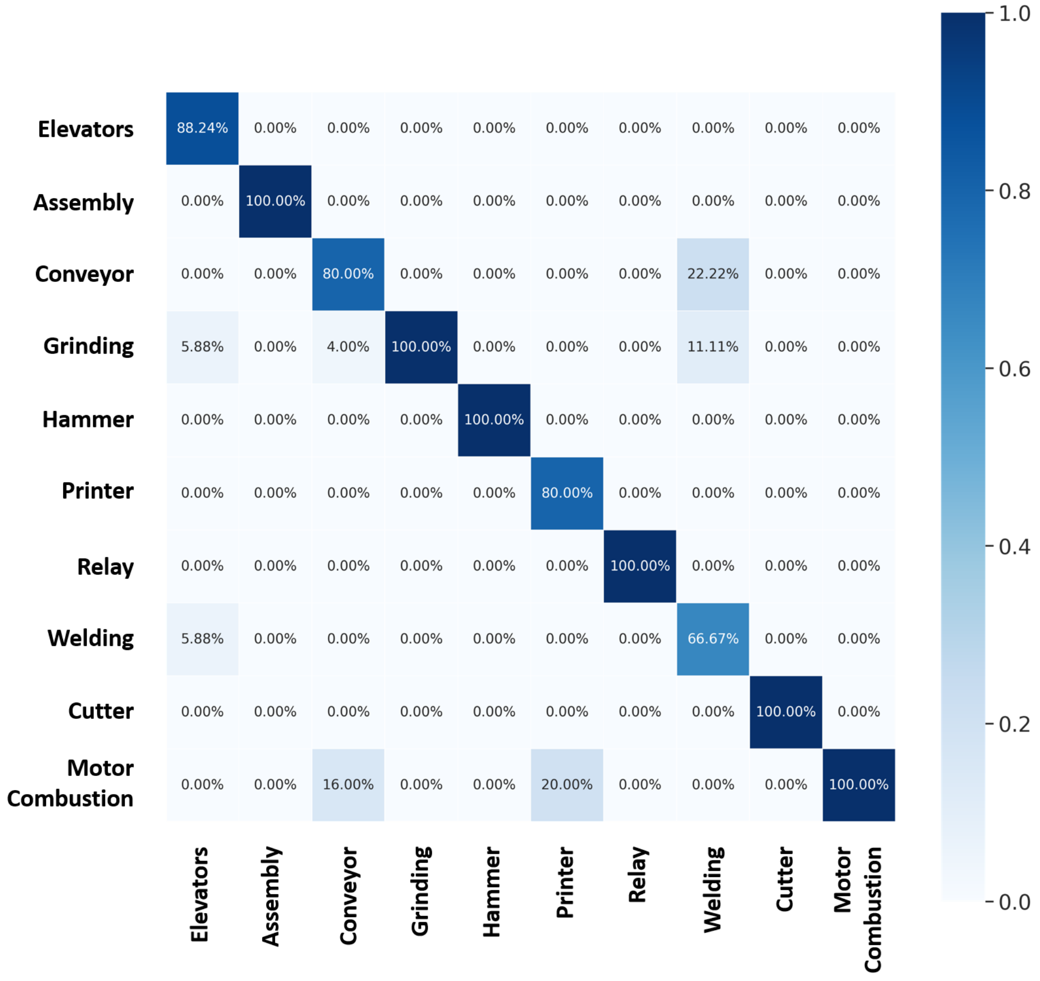

Fig. 7. Confusion Matrix of the Proposed Method on Soundscapes2 Dataset

Transactions on Audio, Speech, and Language Processing, vol. 23, no. 1, pp. 20–31, 2014.

[13] J. X. Yu, K. M. Lim, and C. P. Lee, "Move-cnns: Model averaging ensemble of convolutional neural networks for facial expression recognition." *IAENG International Journal of Computer Science*, vol. 48, no. 3, pp. 519–523, 2021.

[14] Y. S. Tan, K. M. Lim, and C. P. Lee, "Wide residual network for vision-based static hand gesture recognition." *IAENG International Journal of Computer Science*, vol. 48, no. 4, pp. 906–914, 2021.

[15] C. P. Lee and K. M. Lim, "MFRD-80K: A dataset and benchmark for masked face recognition." *Engineering Letters*, vol. 29, no. 4, pp. 1595–1600, 2021.

[16] X. Zhang, M. Li, and C. Huang, "Research on traffic acoustic event detection algorithm based on model fusion." *Engineering Letters*, vol. 29, no. 3, pp. 1078–1082, 2021.

[17] H. Phan, L. Hertel, M. Maass, and A. Mertins, "Robust audio event recognition with 1-max pooling convolutional neural networks," *arXiv preprint arXiv:1604.06338*, 2016.

[18] X. Xia, R. Togneri, F. Sohel, and D. Huang, "Frame-wise dynamic threshold based polyphonic acoustic event detection," 2017.

[19] S. Mun, M. Shin, S. Shon, W. Kim, D. K. Han, and H. Ko, "Dnn transfer learning based non-linear feature extraction for acoustic event classification," *IEICE TRANSACTIONS on Information and Systems*, vol. 100, no. 9, pp. 2249–2252, 2017.

[20] S.-Y. Chou, J.-S. R. Jang, and Y.-H. Yang, "Framecnn: A weakly-supervised learning framework for frame-wise acoustic event detection and classification," *Recall*, vol. 14, pp. 55–64, 2017.

[21] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 279–283, 2017.

[22] D.-N. Jiang, L. Lu, H.-J. Zhang, J.-H. Tao, and L.-H. Cai, "Music type classification by spectral contrast feature," in *Proceedings. IEEE International Conference on Multimedia and Expo*, vol. 1. IEEE, 2002, pp. 113–116.

[23] T. Zhang, J. Liang, and B. Ding, "Acoustic scene classification using deep cnn with fine-resolution feature," *Expert Systems with Applications*, vol. 143, p. 113067, 2020.

[24] D. Kim, S. Park, D. K. Han, and H. Ko, "Multi-band cnn architecture using adaptive frequency filter for acoustic event classification," *Applied Acoustics*, vol. 172, p. 107579, 2021.

[25] T. K. K. Ho and J. Gwak, "Multiple feature integration for classification of thoracic disease in chest radiography," *Applied Sciences*, vol. 9, no. 19, p. 4130, 2019.

[26] U. Zölzer, X. Amatriain, D. Arfib, J. Bonada, G. De Poli, P. Dutilleux, G. Evangelista, F. Keiler, A. Loscos, D. Rocchesso *et al.*, *DAFX-Digital audio effects*. John Wiley & Sons, 2002.

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *European Conference on Computer Vision*. Springer, 2016, pp. 630–645.

[28] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*. PMLR, 2015, pp. 448–456.

[29] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. JMLR Workshop and Conference Proceedings, 2011, pp. 315–323.

[30] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," in *Proceedings of the 22nd ACM International Conference on Multimedia*, 2014, pp. 1041–1044.

[31] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[32] A. Kumar, M. Khadkevich, and C. Fügen, "Knowledge transfer from weakly labeled audio using convolutional neural network for sound events and scenes," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 326–330.

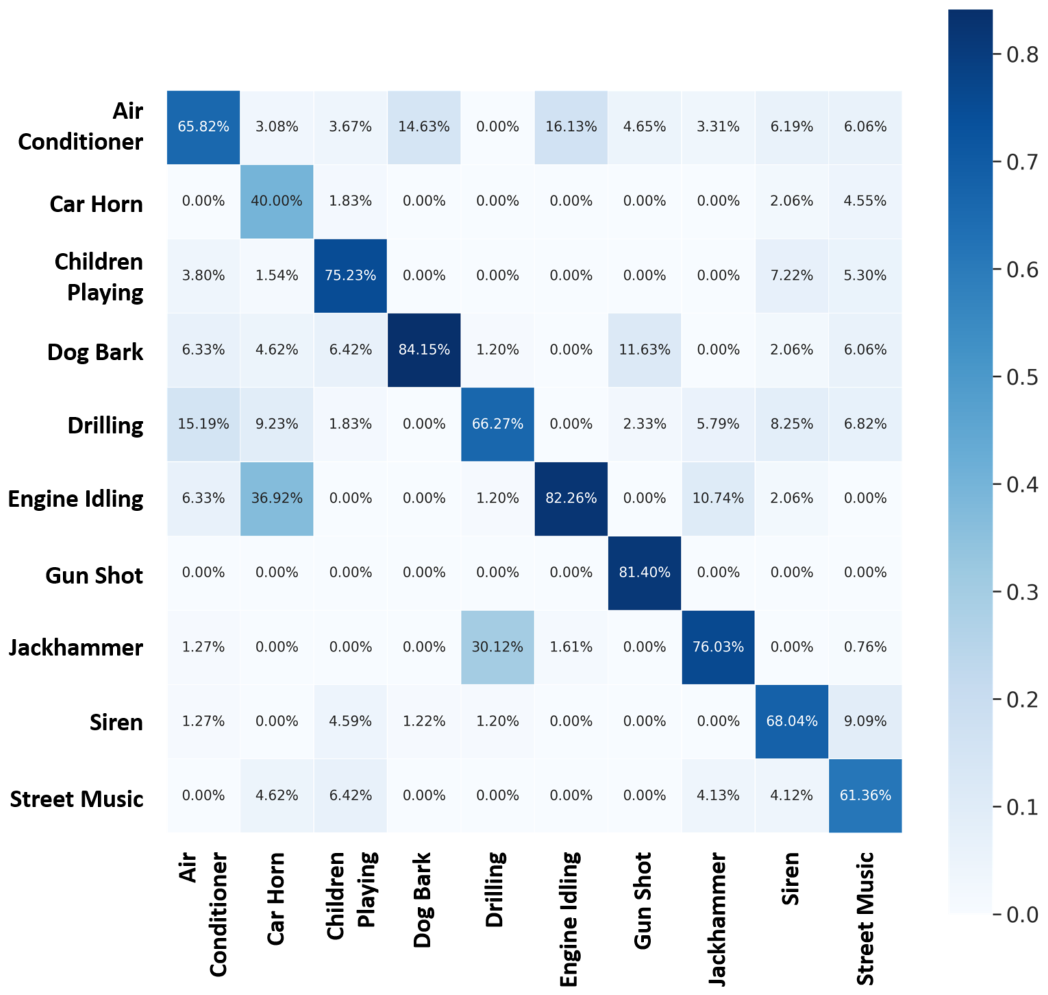

Fig. 8. Confusion Matrix of the Proposed Method on UrbanSound8K Dataset

[33] S. Lee and H.-S. Pang, “Feature extraction based on the non-negative matrix factorization of convolutional neural networks for monitoring domestic activity with acoustic signals,” *IEEE Access*, vol. 8, pp. 122 384–122 395, 2020.

[34] L. Van der Maaten and G. Hinton, “Visualizing data using t-sne.” *Journal of machine learning research*, vol. 9, no. 11, 2008.