DOI: 10.15849/IJASCA.250330.09

Int. J. Advance Soft Compu. Appl, Vol. 17, No. 1, March 2025
Print ISSN: 2710-1274, Online ISSN: 2074-8523

Performance Analysis of Convolutional Neural Networks with Different Optimizers and Hybrid Pooling Methods for Sound Classification

Pratibha Rashmi and Manu Pratap Singh

Department of Computer Science, Dr. Bhimrao Ambedkar University, Agra, UP, India
E-mail: pratibha.rashmi@gmail.com
Department of Computer Science, Dr. Bhimrao Ambedkar University, Agra, UP, India
E-mail: manu_p_singh@hotmail.com

Abstract

Convolutional Neural Networks (CNNs) are used for the sound signal classification successfully due to their capabilities of deep architecture with pooling schemes. Since the pooling methods in CNN work as regularization to minimize the effect of overfitting which further improves the performance of CNN. Various pooling techniques are proposed to enhance the performance of CNN for classification, but mostly all pooling approaches are considered for the classification of images. Sound samples are of a different nature from the static images and the temporal sequences of sounds play a crucial role in the digitalization of sound samples. The presented work of this paper considered the different pooling schemes with convolutional neural networks to improve the classification accuracy for environment sounds. In this approach, two different CNN architectures are implemented and analyzed with various pooling schemes at local and global levels. Further, the paper presented a new pooling scheme at the local and global pooling aspects to improve the performance. This new pooling scheme is considered as max-min difference pooling. The simulated results are obtained on the UrbanSound8k dataset with different optimizers and proposed pooling schemes. The experimental results indicate improved performance of CNN model with max-min difference pooling with global aspect.

Keywords: *CNN, Sound Classification, Stochastic Pooling, Hybrid Pooling, Deep Learning.*

1 Introduction

In most signal processing techniques, sound samples are considered in the form of time-frequency patches. These samples contain original sounds with lots of white noises and other inferences in the sounds. Several machine learning and deep learning techniques have been proposed to get better results for the classification of environmental sounds [1]. An automated environmental sound classification system is used in several areas, like sound investigation and identification, in various applications of surveillance in the public domain. Earlier work considered the pre-trained models of deep neural networks. These

Received 4 January 2025; Accepted 18 March 2025

models were trained for image classification and used on the existing datasets of environment sounds [2]. Generally, the spectro-temporal features are considered for the representation of input feature maps [3]. Environmental sound, unlike regular and structured sounds, lacks both static time patterns and semantic sequences. Therefore, identifying universal features from the sound samples presents a challenge. Besides this, environmental sound contains a significant amount of white noise and unrelated sound, leading to the emergence of complex structures such as variability, diversity, and unstructured characteristics [4, 5]. Various signal processing methods and machine learning techniques have been used to address these issues for environmental sound classification. Initially, the feature extraction and classification problems were considered separate in the form of pre-processing and classification. In these approaches, the feature extraction step has been implemented with manual operations like mel-frequency cepstral coefficient (MFCC), Mel spectrum feature, and wavelet transformation [6]. Further, in the second step, machine learning methods like Support Vector Machine (SVM) [7], K-nearest neighbors [8], matrix factorization [9], and extreme learning algorithms were employed to generate classification from the extracted features. These approaches improved classification performance but consumed lots of time to construct feature representations and to identify the best combination of functions. Deep Neural Networks have been proven to have a strong ability to extract features automatically and effective classification with various deep neural network models [10]. Attention mechanisms have also been applied for sound recognition and speech sentiment analysis [11]. In this approach, the temporal-frequency attention-based convolutional neural network model is proposed on the UrbanSound8k and ESC-50 datasets to show the accuracy of the model for classification [12]. In deep learning models, earlier models utilized Log-Mel and its spectrogram as a two-dimensional feature representation to input into the network for learning and classification [13]. The data augmentation strategies were also used to fulfill the requirement of the huge amount of data to enhance the capabilities of the Deep Neural Network [14]. Further, the original audio waveform as input to train CNN is proposed, and a large number of experiments have been conducted with the number of CNN layers as the independent variable [15]. In another development, an end-to-end classification model based on 1D-CNN is used for direct feature extraction from raw audio waveforms of any length [16]. In the recent development of deep learning for sound classification, the attention mechanism is combined with a deep neural network to calculate the weight metric and automatically assign the corresponding weights for each frame-level feature [17]. Further, the attention mechanism with deep learning is used for the environmental sound classification [18]. Later on, various methods were employed for feature extraction from sound signals, and deep neural networks with deep architecture exhibited better classification performance [19]. The convolution layers in the architecture aim to extract the patterns found within the local region of the input 2D time-frequency patch of sound signals. This can be obtained with many filters on the inputs, computing the inner product of the filters at every location in the input, and outputting these as feature maps. The resulting activations are then passed to the pooling layer and further to the dense network, followed by the classification layer. The accuracy of the classification depends on various parameters. Among them, the pooling schemes have an important role in addressing the overfitting issues of convolutional neural networks. Regularization is considered one of the prominent methods to minimize the overfitting of any arbitrary sample example. Different approaches to regularization, such as dropout and data augmentation of the training set [20], are used. The dropout process [21] stochastically considers half of the activation with a layer to zero during the training process. It significantly improves the performance of large neural networks for classification, but it does not exhibit the same

benefits for convolutional layers. Hence, there is a need to provide a stochastic regularization also for the convolutional layers. Pooling schemes explore the regularization in deep convolutional neural networks and improve the performance of the network for classification. In deep convolutional neural network architectures, the pooling layer is considered after each convolution layer. The pooling layer uses the feature map provided by the kernel of the convolution layer. Commonly, the max and average pooling functions are used in pre-trained CNNs, but these types of conventional pooling schemes exhibit drawbacks. In average pooling, all elements in a pooling region are considered, even if many have low magnitudes, and in max pooling, the problem of overfitting the training set occurs [22]. In the literature of deep learning, various pooling schemes are discussed for CNN to obtain better classification accuracy. It includes hybrid pooling [23], max-out network pooling [24], spatial pooling [25], spatial pyramid pooling [26], fractional pooling [27], rank-based pooling [28], soft pooling, and stochastic pooling [20]. A successful variant of pooling is Lp pooling [29]. Lp pooling can be viewed as a continuous parameterization transition from average to max pooling. Lp pooling has shown a large improvement in error rate in the comparison of max pooling. The stochastic pooling [20] is another improved pooling strategy because it is less prone to overfitting due to its random nature, which picks the activations within each pooling region based on the normalized probability of activation values. Most of the pooling schemes proposed in the literature are considered for the classification of images [30], but the nature of sound samples is different from the images. The temporal sequences of voice play a crucial role in sound identification. Besides this, the background noises in the original sample affect the classification accuracy. Therefore, the spiral pattern with 2D-M4-based pooling is proposed for the environmental sound classification [31]. This method used the statistical moments to generate low, middle, and high-level features. This proposed approach improves the environmental sound classification but increases computation complexity and lots of statistical calculations. Thus, lots of works have been reported for the environmental sound classification using pre-trained convolutional neural network models with different pooling schemes. In most of the architectures, a local pooling criterion is employed after each convolution layer to reduce the size of the feature map produced by the kernel of the convolution layer. In most of the cases, max pooling or average pooling is used for further processing. Further different combinations of both pooling schemes are considered to improve the classification accuracy. Despite all of these attempts and implementations of hybrid pooling schemes, the pre-trained convolutional neural network could not reach the expected accuracy for sound data due to the variability and characteristics of sound samples and its representation of the input feature map of a 2D mel-spectrum image. To keep all these issues in mind, we considered the two different types of convolutional neural networks with different pooling schemes and local and global pooling criteria to analyze the performances of CNN for the classification of environmental sound classification. The main objective of the proposed research work is to design convolutional neural network models with different hybrid pooling schemes to analyze the performance for the classification of environmental sound. In this implementation, first, we considered a Visual Geometry Group Network (VGGNet) convolutional neural network model, which consists of hierarchical convolutional layers with an activation function followed by batch normalization. The pooling layer is employed only after the last convolution layer instead of after each convolution layer. The local pooling is used for the first architecture, followed by the dense network and classification layer. In the second architecture, the same hierarchy has been followed, but local pooling is replaced with the global pooling scheme. In this architecture, the dense network has not been employed. The output feature map of global pooling is directly passed to the classification layer. In both

the pooling criteria, different types of hybrid pooling schemes like a linear combination of max-average pooling, a linear combination of max-min pooling, soft pooling, and stochastic pooling are employed to analyze the performance of convolutional neural networks. In addition to this, stochastic pooling is used with two methods. In the first method, the probability feature map is constructed from the feature map using the multi-nominal distribution formed by the activation with the pooling region, and then the activation is selected using the probability function. In the second method, the activation value is selected on a random basis without drawing the probabilistic map. Beside all these methods, the paper also considers another hybrid approach of pooling named as, max-min difference pooling. The major contributions of the presented work in the paper are as follows:

- Two different convolutional neural network models are designed. These are different from existing pre-trained VGGNet type models.
- Local pooling is applied after the last convolution layer, followed by two dense layers and a classification layer.
- Global pooling layer is considered also after the last convolution layer followed by classification layer. The dense network has been removed from second architecture to obtained better normalization.
- Various hybrid pooling schemes are explored in combination with different stochastic optimization methods.
- A novel local pooling scheme, max-min difference pooling, is introduced to analyze the performance of both architectures for environmental sound classification. This approach reduces overfitting through adaptive weight scaling.
- The proposed CNN architectures with max-min difference pooling optimize the number of parameters and reduce training time with the improvement in computational efficiency.

Thus, the different pooling schemes with local and global pooling criteria are considered to analyse the performance of both the proposed CNN architectures on the UrbanSound8k dataset of environment sounds.

In the experimental process of implementing the proposed Convolutional Neural Networks (CNNs) with hybrid pooling methods for sound classification, several challenges were encountered. Integrating hybrid pooling methods, such as combining max pooling and average pooling, presented a significant challenge due to the need for careful design to prevent information loss and preserve feature representation. To address this, multiple configurations of pooling methods were implemented and thoroughly tested. By evaluating the performance impact of each experiment on the proposed model, it became possible to identify the most effective hybrid pooling scheme, finally enhancing the performance of the proposed model while minimizing any drawbacks related to specific pooling methods. Overfitting is also a big problem, especially when using deep neural network models with limited data. To solve this, we used several tricks. First, we applied regularization techniques like dropout, which help prevent the model from relying too much on specific features. Batch normalization and L1 regularization with a regularization factor of 0.001 are helped to reduce overfitting, making the proposed model more reliable when classifying new, unseen sounds. The selection of a suitable optimizer for sound classification system presented a significant issue, since different optimizers like Stochastic Gradient Descent (SGD), Adaptive Moment Estimation (Adam) and Adaptive

Gradient Algorithm (AdaGrad) exhibit different behaviors based on the data and model architecture. A systematic approach was implemented to deal with this complexity. A series of experiments are conducted and the results of these experiments are then analysed to ensure the most effective optimizer for sound classification task. This systematic process ensured that the chosen optimizer was well-suited for sound classification and improving the performance of the model and training efficiency.

This paper is organized as follows: section 2 of the paper discusses the Material and methods for different pooling schemes which includes architecture framework, pooling methods and experimental design. Section 3 of the paper consist the implementation and simulation design of proposed model of convolution neural network and Section 4 includes the results and discussion of performance analysis with various methods of pooling and state-of-art performance of CNNs for the Urban Sound classification. Finally, the conclusion is presented in the last section followed by the references.

2 Materials and Methods

Generally, convolutional neural networks consist of kernel layers followed by the pooling layers, flatten layers, dense network, and classification layer. In this type of architecture, the input sample is presented to the first kernel layer, which extracts pattern information within a local region of the input sample to compute the inner product between the region of the input sample and associated weights. The generated feature maps of the output of the layer are passed to the pooling layers for down-sampling of the feature map and to learn large-scale features that are invariant to small local transformations. The same operation repeats for the successive convolutional layers. In this paper, we considered different convolutional neural network architectures to incorporate pooling methods more effectively and impactful.

2.1 Proposed Architectural Framework

The proposed architectural frameworks build upon convolutional blocks designed for efficient feature extraction. Each block within the architecture consists of a convolutional layer followed by a Rectified Linear Unit (ReLU) activation function and batch normalization to ensure stability during training. These fundamental blocks are replicated three times, forming the basic structure of the models. These repeated convolutional blocks enhance feature extraction and allow the network to identify complex patterns within the input feature map. After stacking the three convolutional blocks, each convolutional block is followed by a pooling layer to down-sample the feature maps and to learn large-scale sound features that are invariant to small local transformations. Here, two different approaches are used to facilitate the pooling scheme. In the first approach, local pooling is used followed by the flattened layer, dense layers, and classification layer as shown in Fig 1(a), and in the second approach, global pooling is used and eliminates the dense layer. The output of the global pooling is presented to the flattened layer followed by the classification layer as shown in Fig. 1(b). The proposed convolutional neural networks are shown below Figs.

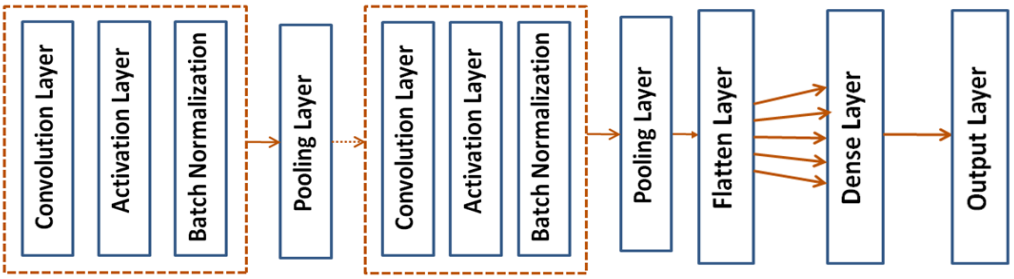


Figure 1(a): First Proposed Convolution Neural Network Architecture with Local Pooling

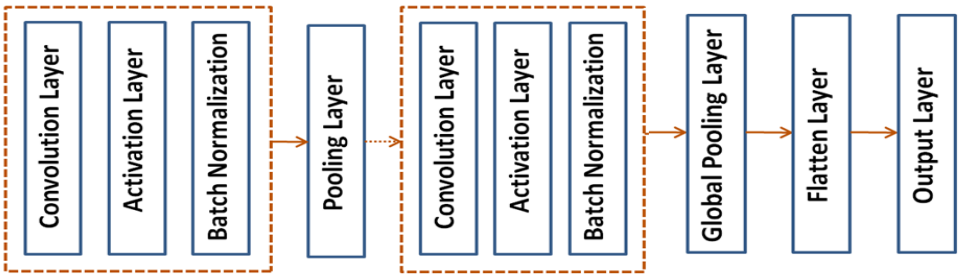


Figure 1(b): Second Proposed Convolution Neural Network Architecture with global pooling

In our proposed implementations, various types of pooling are utilized, namely max pooling, average pooling, linear combination of max-average, max-min, soft pooling, and stochastic pooling. After all convolutional blocks, the CNN model is flattened to convert the multi-dimensional feature maps into a one-dimensional feature vector, which is more suitable for the classification layers. The first proposed model contains two dense layers. The first dense layer has 128 units, activated by the ReLU activation function, and incorporates a dropout layer with a rate of 0.20 to mitigate overfitting. The second dense layer follows with 64 units, using the same activation function and dropout setup. Both dense layers also include L1 regularization to prevent overfitting.

We chose 128 units for the first dense layer because the flatten layer produces a feature map with dimensions 128×1 for each pattern. The second dense layer consists of half the units of the first dense layer, i.e., 64 units. The selection of the number of units for the dense layers is based on the dimensionality of the feature map from the flatten layer, with the second layer containing half the units of the first dense layer to achieve a balanced downscaling. The second proposed model contains no dense layer. The output of the global pooling layer passes to the flatten layer and is directly presented to the output layer. Finally, the output layer with 10 units, corresponds to the number of classes and employs the softmax activation function to produce probabilities for each class. The implementation details with hyperparameters for the first and second CNN architectures can be shown in Table 1 and Table 2 respectively.

Table 1: Implementation details with hyper parameters for the first architecture

So. No.	**Layer Type**	**# of Filters**	**Kernel Size**	**Pool Size**	**Activation function**	**Parameters**	**Output Shape**	**Regularization/Dropout**
1.	Conv2D 1	128	2x2		ReLU	640	(None,64,64, 128)	
2.	BatchNorm					512	(None,64,64, 128)	
3.	Pooling			2x2			(None,32,32,128)	
4.	Conv2D 2	64	2x2		ReLU	32,832	(None,16,16,64)	
5.	BatchNorm					256	(None,16,16,64)	
6.	Pooling			2x2			(None,8,8,64)	
7.	Conv2D 3	32	2x2		ReLU	8,224	(None,4,4,32)	
8.	BatchNorm					128	(None,4,4,32)	
9.	Pooling			2x2			(None,2,2,32)	
10.	Flatten						(None, 128)	
11.	Dense	128			ReLU	16,512	(None, 128)	L1 Regularization (λ=0.001)
12.	Dropout						(None, 128)	Dropout Rate: 20%
13.	Dense	64			ReLU	8,256	(None, 64)	L1 Regularization (λ=0.001)
14.	Dropout						(None, 64)	Dropout Rate: 20%
13.	Dense	10			Softmax	650	(None, 10)	
Total Params: 68,010								
Trainable Params: 67,562								

Table 2: Implementation details with hyperparameters for the second architecture

So. No.	**Layer Type**	**# of Filters/ Units**	**Kernel Size**	**Pool Size**	**Activation function**	**Parameters**	**Output Shape**	**Regularization/Dropout**
1.	Conv2D 1	128	2x2		ReLU	640	(None,64,64, 128)	
2.	BatchNorm					512	(None,64,64, 128)	
3.	Pooling			2x2			(None,32,32,128)	
4.	Conv2D 2	64	2x2		ReLU	32,832	(None,16,16,64)	
5.	BatchNorm					256	(None,16,16,64)	
6.	Pooling			2x2			(None,8,8,64)	
7.	Conv2D 3	32	2x2		ReLU	8,224	(None,4,4,32)	
8.	BatchNorm					128	(None,4,4,32)	
9.	GlobalPooling						(None, 32)	
10.	Flatten						(None, 32)	
11.	Dropout						(None, 32)	Dropout rate: 20%
12.	Dense	10			Softmax	330	(None, 10)	L1 Regularization (λ=0.001)

Total Params: 42,922
Trainable Params: 42,474

2.2 Pooling Methods

It is well-considered that pooling does not only decrease the size of the receptive field of convolutional kernels over layers but also reduces the computational complexity and memory requirements as it reduces the resolution of the feature map while preserving important features that are needed for processing by the subsequent layers [32]. Thus, the pooling is an important step in convolutional-based systems that reduce the dimensionality of the feature maps. The pooling operators provide a form of spatial transformation invariance as well as reduce the computational complexity for upper layers by eliminating some connection between convolutional layers. This layer executes the down-sampling on the feature maps coming from the previous layers and produces the new feature maps with a condensed resolution. Generally, this layer has two main purposes, the first is to reduce the number of parameters or weights, thus reducing the computational cost and the second is to overcome the problem of overfitting. Normally, there are two groups of pooling used for the convolutional neural networks. The first one is local pooling in which the pooling is performed from small local regions to down-sample the feature maps. The second one is global pooling, which is performed from each of the entire feature maps to get a scalar value of a feature vector [33]. From the viewpoint of sound signals, the pooling is a form of non-linear down-sampling [34]. Therefore, for the environmental sound signals both the local and global pooling are considered for effective classification of the sound signals.

Convolution neural networks are widely used for the classification of environmental sound signals with different pooling methods. Generally in all these CNN architectures, environment sound signals are considered as the 2D Time-Frequency patches and presented as input to the CNN. A powerful CNN is composed of several feature extraction stages and each stage consists of a convolution layers, a non-linear transformation layer, and a pooling layer. The convolution layer considers the inner product of the linear filter and the underlying receptive field followed by a nonlinear activation function at every local portion of the input. Thus, in this layer, the output feature map can be represented as [38]:

$$y_k = f(w_k \otimes x) \quad (1)$$

Where, x denotes the input vector, w_k is the associated convolution filter with the k^{th} feature map, $\otimes$ indicates the convolution operator and $f(\cdot)$ is the non-linear activation function. Usually, there are two kinds of non-linear transformations. One is the local response normalization, which yields the normalized output $y_{k_{ij}}$ at the location (i, j) in k^{th} feature maps as [40]:

$$y_{k_{ij}} = \frac{x_{k_{ij}}}{\left(1+\frac{\alpha}{N}.\sum_{l=k-\frac{N}{2}}^{k+\frac{N}{2}}\left(x_{lij}\right)^2\right)^{\beta}} \quad (2)$$

Where the sum runs over N adjacent feature maps at the same spatial location and the parameters of α and β can be determined by using a validation set.

Another is the local contrast normalization [41] with the normalized output $y_{k_{ij}}$ as be computed as:

$$y_{k_{ij}} = \frac{x_{k_{ij}}}{\left(1+\frac{\alpha}{M_1 M_2}.\sum_{p=i-\frac{M_1}{2}}^{i+\frac{M_1}{2}}\sum_{q=j-\frac{M_2}{2}}^{j+\frac{M_2}{2}}\left(x_{k_{pq}}-m_{k_{ij}}\right)^2\right)^{\beta}} \quad (3)$$

Here, the local contrast is computed within a local $M_1 \times M_2$ region with the center at (i, j). $x_{k_{pq}}$ is the element at (p, q) within the pooling region and $m_{k_{ij}}$ is the mean of all x values within the $M_1 \times M_2$ region is the k^{th} feature map as [41]:

$$m_{k_{ij}} = \frac{1}{M_1 M_2} \sum_{p=i-\frac{M_1}{2}}^{i+\frac{M_1}{2}} \sum_{q=j-M_2/2}^{j+M_2/2} x_{k_{pq}} \quad (4)$$

The different feature maps are generated from different filters of the convolution layer. The generated output y_k further passes to the pooling layer and produces the pooled feature map. Let a set of feature maps is F and a pooling region P defined on one of these feature map i.e. F_k. Let $x \in \mathbb{R}^{W' \times H'}$ represents the features that are inside the pooling region P on F_k. It covers both the local and global pooling. If $W = W'$ and $H = H'$ where W' & H' are pooling region then it is a global pooling and if $W' < W$ and $H' < H$ then it shows the local pooling. These W & H represent the width and height of the feature map respectively.

In most of the convolution neural network models, the max and average pooling are used. The max pooling selects the largest element in each pooling region as [32]:

$$f_{max}(x) = \max_{x_i \in \mathbb{R}} (x_i)_{i=1}^{N} \quad (5)$$

Fig. 2 is representing an example of exhibiting the max pooling scheme.

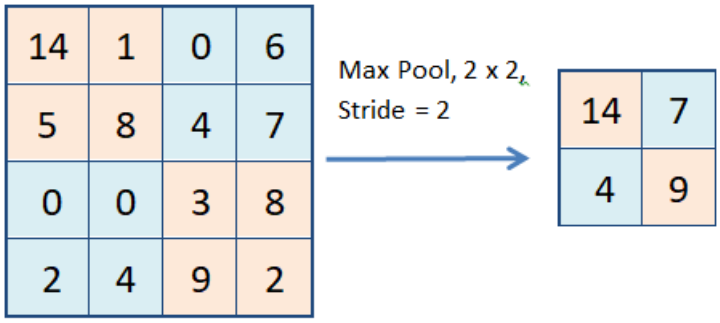


Figure 2: Conceptual representation of max pooling

In average pooling the arithmetic sum of the elements in each pooling region is considered as [32]:

$$f_{avg}(x) = \frac{1}{N} \sum_{i=1}^{N} |x_i|_{x_i \in x} \quad (6)$$

here $N = W.H$ is the number of elements in x i.e. x_i is the i^{th} element of x where $i = 1, 2, \ldots \ldots N$. Fig. 3 is representing an example of exhibiting the average pooling scheme.

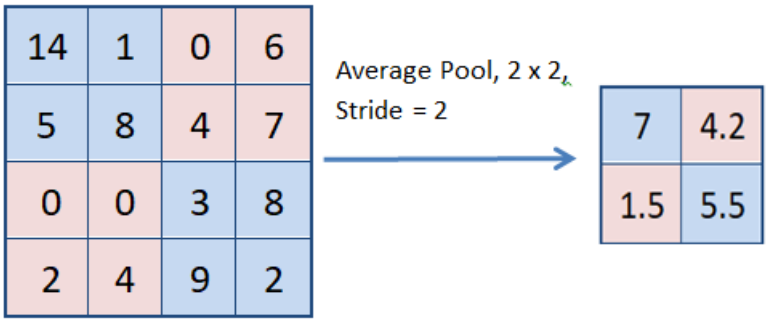


Figure 3: Conceptual representation of average pooling

Max pooling and the average pooling have their advantages and disadvantages. During the training of a deep convolution network, in the average pooling, all elements in a pooling region are considered, even if many have low magnitude, and in the max pooling the problem of overfitting the training set occurs [20]. In the sound dataset, the feature map is usually represented in the two-dimension time-frequency patches. These patches also contain white noise due to which the local pooling and global pooling methods exhibit the

data over the training of a high rate. Mostly, the CNNs use the average pooling and max pooling [37] due to their simplicity but they do not have parameters to tune. Another problem with the sound signal is overlapping regions with actual sound and background noise with the existence of white noise. The log-mel spectrum or MFCC feature map of the sound samples is generally considered as the 2D image. These images include the features of actual sounds and noises. Thus the feature images of the sound samples are of the ever changing nature. Thus, it is of high probability that the defective aspects of max and average pooling will exhibit negative effects in applying pooling layers to CNNs. Therefore, the other methods of the pooling are considered to apply as a solution. In this attempt, these deterministic pooling operations can be replaced with stochastic procedures or some hybrid approaches.

Hence, with the deterministic pooling operation, the mixed pooling is considered. In this hybrid approach, the max-pooling and average pooling is combined linearly as [38]:

$$f_j = \lambda \max_{i \in R_j} x_i + (1 - \lambda) \frac{1}{|R_j|} \sum_{i \in R_j} x_i \qquad (7)$$

Where λ decides the choice of either using max pooling or average pooling. The value of λ is selected randomly and it is recorded for forward-propagation order and it is used during the back-propagation process. Fig. 4 represents an example of exhibiting the linear combination hybrid pooling schemes.

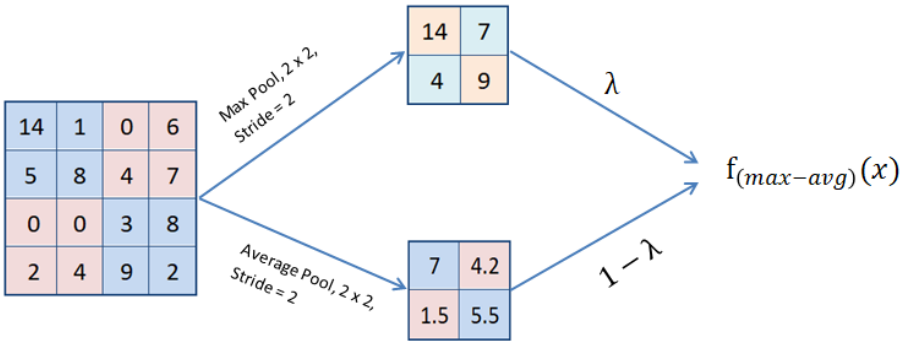


Figure 4: Conceptual representation of linear combination of max-average pooling

This hybrid approach can be further extended with the linear combination of max-min pooling as:

$$f_j = \mu \max_{i \in R_j} x_i + (1 - \mu) \min_{i \in R_j} x_i \qquad \text{with } 0 \le \mu \le 1 \qquad (8)$$

Here, again the value μ is assigned randomly in between 0 and 1. It is also used during the back-propagation process. Fig. 5 represents an example of exhibiting the max-min hybrid pooling schemes.

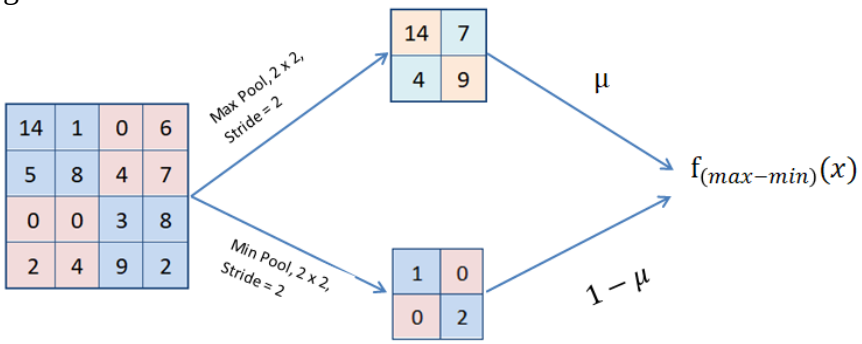


Figure 5: Conceptual representation of linear combination of max-min

Another hybrid approach is considered as Lp pooling [39]. This approach claims that its generalization ability is better than max pooling. In this pooling approach, a weighted average of inputs is taken in the pooling region and can be expressed as [39]:

$$f_j = \left(\frac{1}{|R_j|}\sum_{i\in R_j} x_i{}^p\right)^{1/p} \quad (9)$$

Where, f_j represents the output of the pooling operator at location j, x_i is the feature value at location i within the pooling region R_j. The value of p varies between 1 and ∞, and we keep $p > 1$ to set the trade-off between average and max pooling. The 2D images of sound samples contain variations due to white noise or variations in the voice pitches. The mel-log or MFCC spectrum of sound samples carries actual words with a huge amount of noise. The 2D images of these spectrums also carry the same. The region of the input set is to identify the actual sample from the images. Thus, it is required to localize the area of the actual voice in the 2D image. Therefore, in that case, the mixing proportion should depend on the characteristics of each image rather than the characteristics of the actual voice data. Thus, the gated max-average pooling [46] is considered to address these issues as:

$$S(x) = \sigma(w^T x)S_{max}(x) + \left(1 - \sigma(w^T x)\right)S_{avg}(x) \quad (10)$$

Where, $w \in R^n$ is a weight vector considered as a "gating mask" to be learned when training the network and $\sigma(.)$ is a sigmoid function [40]. In this approach, the parameters can also be learned separately for each layer or separately for each of the channels in each layer of the network. Another method for combining the max pooling and average pooling is considered in the form of soft pooling approach [41]. It is used as an intermediate form between max and average pooling. In this approach, a smooth differentiable function is used to approximate the max and average pooling for different parameters settings as [41]:

$$S_{(sp)} = \left(\frac{1}{N}\sum_{i=1}^{N}|x_i|^r\right)^{1/r} \quad (11)$$

Where, the parameter r controls the softness. Although various methods of pooling are proposed with different combinations of max and average pooling, in all such methods the issue of overfitting occurs during the training with CNN. The stochastic pooling is proposed to reduce overfitting by introducing randomness in the pooling procedure. Therefore, inspired by the dropout techniques, Zeiler and Fergus [20] proposed the idea of stochastic pooling. The stochastic pooling applies multinomial distribution to pick the value randomly. In this process, first probabilities are computed for each region by normalizing the activations within the regions, as [46]:

$$p_i = \frac{a_i}{\sum_{k\in R_j} a_k} \quad (12)$$

These probabilities create a multinomial distribution that is used to select location l and corresponding pooled activation a_l based on p. Multinomial distribution selects a location l within the region as:

$$s_j = a_l \text{ where } l \sim P(p_1, \dots, p_{|Rj|}) \quad (13)$$

Thus, the activations are selected based on the probabilities calculated by multinomial distribution. In this, all activations get the chances according to their probability proportionate. In this process, we have the set of k possible results $a_1, a_2, \dots \dots a_k$ with the associated probabilities $(p_1, p_2, \dots \dots \dots p_k)$ i.e. $\sum p_k = 1$.

Here, the P represents the probability function that constructs the index of the region of the filter, and the corresponding activation value is selected with $s_j = a_l$. The probability function P, which is used for the selection of index from the probability of the each activation as:

$$P(X_1, X_2, \dots \dots \dots \dots X_n) = \frac{N!}{\prod_{i=1}^{n} x_i}\prod_{i=1}^{n} P_i(x_i) \quad (14)$$

Where, x_i and non-negative and $\sum_{i=1}^{n} x_i = N$ with $\sum_{i=1}^{n} P_i = 1$

Hence, the probability map selects the activation value from the feature map. Let the feature map from the convolution layer is separated into M × N non-overlaping blocks, where each block region is with the size of $r_1 \times r_2$, Now we consider the m^{th} row and n^{th} column block as.

i.e., $B_{m,n} = \left[b_{m,n}(r_1, r_2), r_1 = 1, \ldots\ldots, R_1 \text{and } r_2 = 1, \ldots\ldots, R_2\right]$ (15)

Where, $1 \leq m \leq M, 1 \leq n \leq N$

The stochastic pooling provides a way to overcome with the disadvantages of the conventional pooling methods [42]. It has a chance to reserve the greatest values and reduces the occasion rates of overfitting. It generates the probability map $S_{m,n}(r_1, r_2)$ as [42]:

$$S_{m,n}(r_1, r_2) = \frac{b_{m,n}(r_1, r_2)}{\sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} b_{m,n}(r_1, r_2)} \quad (16)$$

Where $\sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} b_{m,n}(r_1, r_2) = 1$

After that the random value is created in i.e. $R = (r_1, r_2)$ (17)

It considers the discrete probability distribution for representing the probabilities of activation values from the feature map. Fig. 6 is representing an example of exhibiting the stochastic pooling scheme.

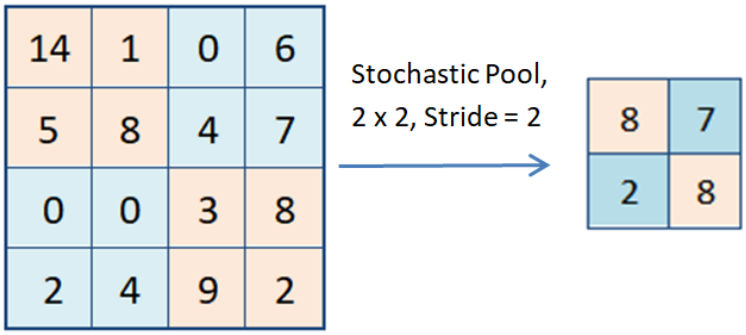


Figure 6: Conceptual representation of stochastic pooling

After analyzing all existing hybrid pooling methods and considering their limitations, we propose a different pooling scheme named max-min difference pooling. In this pooling scheme, we compute the difference between the gated max and min values of the pooling window. Max-min difference pooling can be expressed as:

$$f_s = \sigma(w^T x)[\max(x_i) - \min(x_i)] \quad (18)$$

Where $w \in \mathbb{R}^n$ is a weight vector that is learned during network training, and $\sigma(\cdot)$ is the sigmoid function. In this pooling method, an adaptive weight is considered, which is updated during training and scales the difference between the maximum and minimum values of the pooling window to obtain the feature map.

The advantage of this pooling method is that it reduces overfitting while optimizing parameters during each epoch.

2.3 Experimental Design

In this present work, we used the UrbanSound8k dataset for classification of environmental sounds to provide the training to the proposed Convolutional Neural Networks. UrbanSound8k dataset is containing 8732 labelled sound audios (≤ 4s). These audios come from ten urban sound classes including Air conditioner (AI), Car horn (CA), Children playing (CH), Dog bark (DO), Drilling (DR), Engine idling (EN), Gunshot (GU), Jackhammer (JA), Siren (SI), and Street music (SM). In UrbanSound8K, the occurrences are limited to a maximum duration of 4 sec. To avoid wide variability of class distributions,

the designers limit the number of clips per class to 1000, resulting in 8732 labeled clips. The audio clips are available in .wav format, and the corresponding metadata file is in .csv format. The dataset is split into two sets, on for training set which containing 6985 audio samples, and rest 1747 audio samples for testing. The class distribution of the UrbanSound8k dataset can be shown in Table 3.

Table 3: Class distribution of UrbanSound8K dataset

Class name	Air conditioner [AI]	Car horn [CA]	Children playing [CH]	Dog bark [DO]	Drilling [DR]	Engine idling [EN]	Gun shot [GU]	Jackhammer [JA]	Siren [SI]	Street music [ST]
Label	1	2	3	4	5	6	7	8	9	10
Size	1000	429	1000	1000	1000	1000	374	1000	929	1000

Mel Frequency Cepstral Coefficient (MFCC) is the most widely used feature extraction scheme for speech recognition and audio classification due to its effectiveness in capturing the essential characteristics of audio sounds. Thus, in the proposed work, the MFCC method has been employed to extract features from urban sound samples. Typically, the audio data considered is sourced from existing datasets, consisting mostly of clean samples. Consequently, the spectrogram method is also utilized for feature extraction, transforming the audio data into time-frequency patches. During the feature extraction process, audio data undergoes pre-processing, and it involves sampling, quantization, pre-emphasis processing, and windowing, to convert analog audio signals into a sequence of audio frames. Subsequently, a log-scale Mel spectrogram is utilized to represent the pre-processed audio data as time-frequency patches. The ceptral features are extracted from the mel scaling frequency domain. The sound signal passes through and emphasizes where it will enhance the signal energy at a higher frequency. Now, there exists framing of the digital signal within a frame size of 20-30 ms, featuring either one-half or one-third overlap. The frames will be multiplied by a hamming windowing function to maintain continuity by reducing side lobs. The Fast Fourier Transform will transform the signal into the frequency domain to obtain the magnitude frequency response of the signal frames. A set of mel-filters are applied to the magnitude spectrum to emphasize certain frequency bands. The mel-frequency $M(F')$ related to the common linear frequency F can be considered as [44]:

$$M(F') = 2595 * \log(1 + \frac{F}{700}) \quad (19)$$

A discrete cosine transform is applied on the log energy of the signal to generate different mel-scale cepstral coefficients. So finally, The DCT converts the signal back into a time domain as:

$$C_m = \sum_{j=1}^{N} \cos\left[\pi m * \frac{(K-0.5)}{N}\right] * E_j \quad (20)$$

Where, m is the index of the coefficient and N is the number of mel-filter outputs, C_m are the MFCCs for the frame. E_j logarithm of the Mel-filtered values. Thus, two-dimensional feature vectors in the form of TF-patches are used as input to the proposed convolutional neural networks architectures as shown in the Fig. 7.

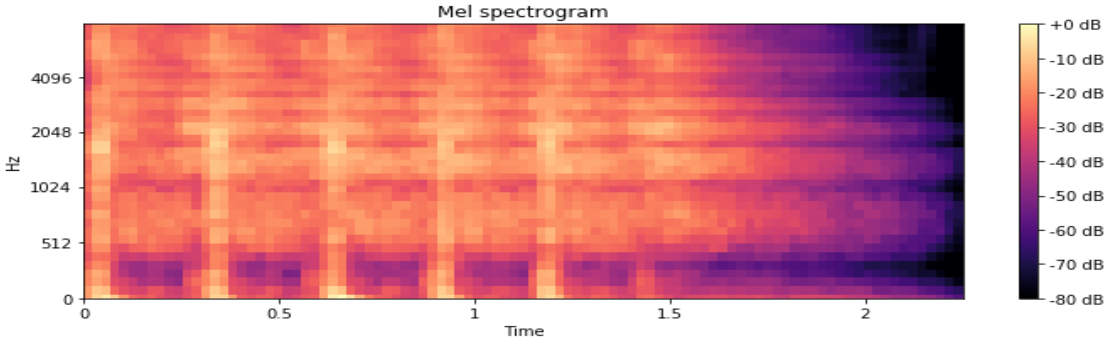


Figure 7: Spectrogram for Siren

Therefore, two-dimensional feature vectors, in the form of time-frequency patches, are employed as input to the proposed convolutional neural network architectures.

3 Implementation and simulation Design

In this present work, we used the UrbanSound8k dataset for classification of environmental sounds to provide the training to the proposed Convolutional Neural Network. In the proposed architecture, we have convolutional blocks. In each block, the convolutional layer consists of k kernels, each with a size of N×N, followed by batch normalization and an activation function. The fundamental operation in each convolutional neural network involves considering the input weight and the number of kernels. Let consider the input matrix as (I_p), kernels as $(K_1, K_2, \dots \dots \dots, K_J)$, and the output as O, with their respective sizes and the S defined as follows:

$$S(x) = \begin{cases} W_1 \times X_1 \times N_I & x = I \\ W_K \times X_K \times N_K & x = K_j (j = 1, \dots J), \\ W_O \times X_O \times N_O & x = O \end{cases} \quad (21)$$

Here, (W, X, N) represent the size of height, width and channels of the matrix respectively. The subscript I, K and O represent input, kernel, and output, respectively. J denotes total number of filters i.e., the channel of input N_I should equal the channel of kernel N_K, and the channel of output N_O should equal to the number of filters J.

Let these filters move with padding of p and stride of q, now we can get the relationship as:

$$W_O = 1 + \frac{(2 \times p + W_I - W_K)}{q} \quad (22)$$

$$X_O = 1 + \frac{(2 \times p + X_I - W_K)}{q} \quad (23)$$

The output of channels will pass to the batch normalization and then activation function. The final output can be represented as:

$$f_1(X) = \text{Pool}(\text{BN}\left(\text{ReLU}\big(S(X)\big)\right)) \quad (24)$$

Now this output works as the input for next convolutional block. In the next convolution block we have the same structure as previous one but the number of kernels is reduced with $J - 2^r$ and the size of the filters are also reduced with $N - 2^{r_1}$, where $1 < r < {}^{J}/_{2}$ and $0 < r_1 < N$. The output of next convolution can be expressed as:

$$f_2(X) = \text{Pool}(\text{BN}(\text{ReLU}\big(f_1(X)\big))) \quad (25)$$

Similarly for the successive blocks we have,

$$f_3(X) = \text{Pool}(\text{BN}(\text{ReLU}\big(f_2(X)\big))) \quad (26)$$

…..

$$f_N(X) = \text{Pool}(\text{BN}(\text{ReLU}\big(f_{N-1}(X)\big))) \quad (27)$$

The sequence of operations i.e., convolutional layer filter, batch normalization and hybrid pooling is continue to each convolution block. Here, we are using different pooling methods to evaluate & analyse the performance of CNN. Now the activation map $f_N(X)$ passes through batch normalization and then the pooling layer for non-linear down-sampling. The pooling layer (PL) provides the invariance-to-translation property to the $f_N(X)$. Hence to a $N \times N$ region, suppose the pixels within the region $\overline{\varphi}$ are:

$$\overline{\varphi} = \begin{bmatrix} \varphi_{1,1} & \varphi_{1,2} & \cdots & \varphi_{1,N} \\ \vdots & \vdots & & \vdots \\ \varphi_{N,1} & \varphi_{N,2} & \cdots & \varphi_{N,N} \end{bmatrix} \quad (28)$$

Now we compute the mean value in the region and obtain the output Z as:

$$Z_{\overline{\varphi}}^{A} = \text{mean}(\overline{\varphi})$$

Further, the max pooling operates on the region $\bar{\varphi}$ and selects the max value as:

$Z_{\bar{\varphi}}^{M} = \max(\bar{\varphi})$

Hence, the average pooling and max pooling were applied on the feature map obtained by the last convolution block. Next, the linear combination of max pooling and average pooling is applied as:

$$Z_{\bar{\varphi}}^{L} = \alpha.\,\text{mean}(\bar{\varphi}) + (1-\alpha)\max(\bar{\varphi}) \quad (29)$$

Further, the linear combination of max-min pooling is obtained from the feature map as:

$$Z_{\bar{\varphi}}^{mm} = \alpha.\min(\bar{\varphi}) + (1-\alpha)\max(\bar{\varphi}) \quad (30)$$

From the equation (26) and (27) the hybrid average pooling can also be obtained as:

$$Z_{\bar{\varphi}}^{h} = \frac{Z_{\bar{\varphi}}^{L} + Z_{\bar{\varphi}}^{mm}}{2} \quad (31)$$

Here, the hybrid average pooling ($Z_{\bar{\varphi}}^{h}$) is used to down-sample the feature map and passes it to the dense network for the classification. The stochastic pooling is introduced to conquer the down-sampling, overfitting and lack of generalization problem due to average and max pooling. In the proposed approach, instead of computing the average or the max, the output of the stochastic pooling (Z^{SP}) is calculated via sampling from a multinomial distribution generated from the activation of each region $\bar{\varphi}$.

Here, in this process first the probability is estimated for each $\varphi_{i,j}$ within the region $\bar{\varphi}$ as [42]:

$$P_{i,j} = \frac{\varphi_{i,j}}{\text{sum}(\bar{\varphi})} \quad (32)$$

or, $P = \frac{\bar{\varphi}}{\sum(\bar{\varphi})}$

Now, we select a location β within $\bar{\varphi}$ in accordance with the probability $(P_{i,j})$ as:

$$\beta \sim \text{Prob}\left(P_{1,1}, P_{1,2}, \ldots\ldots\ldots P_{1,N}, P_{N,1}, \ldots\ldots\ldots P_{N,N}\right) \quad (33)$$

Here, from the equation (33) the random value of the probability is selected. The selected probability value is mapped to region $\bar{\varphi}$ for the selection of output activation value from the feature map as:

$$Z_{\bar{\varphi}}^{SP} = \varphi_{\beta} \quad (34)$$

Therefore, the stochastic pooling uses non-maximal activation from the region $\bar{\varphi}$, instead of outputting the greatest value.

In our proposed approach, after each block of convolutional layer, the batch normalization is used. It means that the batch normalization is actually doing the internal covariant shift. Thus, the batch normalization is used to normalize the internal layer's input $I = \{x_i\}$ over every mini-batch of size m, in order to guarantee the batch normalized output $S = \{y_i\}$. Therefore, to compute the normalized input pattern the following computation is considered.

The empirical mean μ and empirical variance σ^2 over the training set I is obtained as:

$$\mu_I = \frac{1}{m}\left(\sum_{i=1}^{m} x_i\right) \quad (35)$$

$$\sigma_A^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_I)^2 \quad (36)$$

The input $x_i \in I$ is normalized to $\hat{x}_i$ as:

$$\hat{x}_i = \frac{(x_i - \mu_I)}{\sqrt{\sigma_A^2 + \alpha}} \quad (37)$$

Where, α is a constant and it is used to enhance the stability. In the proposed models, we considered $\alpha = 10^{-5}$. The sequence of operations i.e., convolutional layer filter, batch normalization and hybrid pooling is continue to each convolution block. Therefore, in order to have more expressive deep neural network. Thus, the transformation can be usually carried out as: $S_i = \delta \times \hat{x}_i + D, i = 1, 2, \ldots\ldots, m$ (38)

Here, the parameters δ and D are the two learnable parameters during learning. The transformed output $S_i \in S$ is then passed to the next layer and the normalized remains internal to current layer.

Now, the output (S_L) of last convolutional block is presented to the dense network or fully connected network and the output of second dense layer is presented to the classification layer. The network is trained using SGD with mini-batch size of m at a fixed constant momentum term value of α_i and with the loss function on E as:

$$E = -\sum_i \sum_k S_k^{\alpha}(t) \log\left(S_k(t)\right) \quad (39)$$

Where, $S_k^{\alpha}(t)$ and $S_k(t)$ are the target and predicted value of t^{th} training example at k^{th} class respectively. The update rule for $w_{i,j}^{l}$ can be computed as:

$$w_{i,j}^{l}(t) = w_{i,j}^{l}(t-1) + Dw_{i,j}^{l}(t) \quad (40)$$

$$\text{and, } Dw_{i,j}^{l} = \alpha_1 . Dw_{i,j}^{l} - \eta \frac{\partial E}{\partial w_{i,j}^{l}} \quad (41)$$

here, $w_{i,j}^{l}$ represents the weight between $H_i^{l-1}$and H_j^{l}, $\partial w_{i,j}^{l}$ denotes the gradient of $w_{i,j}^{l}$ and η represent the leaning rate. Hence, the proposed CNN architecture is trained with SGD for the fixed number of epochs for each mini-batch and it generate the desired classification for the given training set of environment dataset.

4 Results and Discussion

The simulated results have obtained on the UrbanSound8k dataset for the classification of environment sound with different pooling scheme to evaluate and analyzing the performance of both the CNN architectures. In our previous work [45], we compared different state-of-the-art CNN architectures using various feature extraction and augmentation approaches. A detailed comparative study of all the existing networks has been presented. The observed accuracy for the given dataset, using these methods with different feature extraction techniques, ranged from 73% to 82%. As, in our first proposed Convolutional Neural Network architecture, we considered different local pooling layers i.e. max pooling, average pooling, linear combination of max-average pooling, linear combination of max-min pooling, soft pooling, stochastic pooling, and max-min difference pooling. Basically, local Pooling layers in Convolutional Neural Networks (CNNs) serve the purpose of down-sampling or reducing the spatial dimensions (width and height) of the input patterns whereas in second proposed CNN architecture, we employed global pooling for dimensionality reduction and feature aggregation. The simulated results indicate that local pooling outperforms global pooling in term of sound classification accuracy. By reducing the dimensionality of the feature maps, global pooling accelerates the training process and reduces the overall computational load. Beside these pooling a newly type max-min difference pooling is also introduced to evaluate the performance of CNN. In this pooling scheme, the maximum activation within the pooling window is selected, as well as the minimum activation is also selected within the same window. Subsequently, the difference between the maximum and minimum activations is calculated, and this difference value is considered to select the activation for pooling. In the simulation work, we considered the seven different experiments for each CNN model with different pooling schemes to analyze the performance of CNN for classification of environment sounds. These experiments were conducted on the existing dataset sound samples.

In the simulation design, we considered two optimizers, namely SGD and Adam [20], along with the categorical cross-entropy loss function. Adam is a variant of stochastic gradient descent that dynamically adjusts the step size for each dimension. Both mini-batch SGD and Adam optimizers have been used in various pre-trained CNN architectures

and have demonstrated their effectiveness for learning in terms of optimizing parameters and computational cost. We applied both optimizers to the proposed architectures. The simulation results indicate that the performance of Adam with first CNN architecture is more effective than that of the SGD optimizer. Therefore, we used the Adam optimizer for the final classification results. The performance evaluation of both CNN architectures was conducted with the Adam optimizer and is presented in Fig. 10. Additionally, the confusion matrix is shown for the Adam optimizer with our proposed pooling method, and Fig. 13 presents the performance analysis for the Adam optimizer with the proposed pooling schemes. The performance of first CNN model for classification accuracy is presented in Fig. 8.

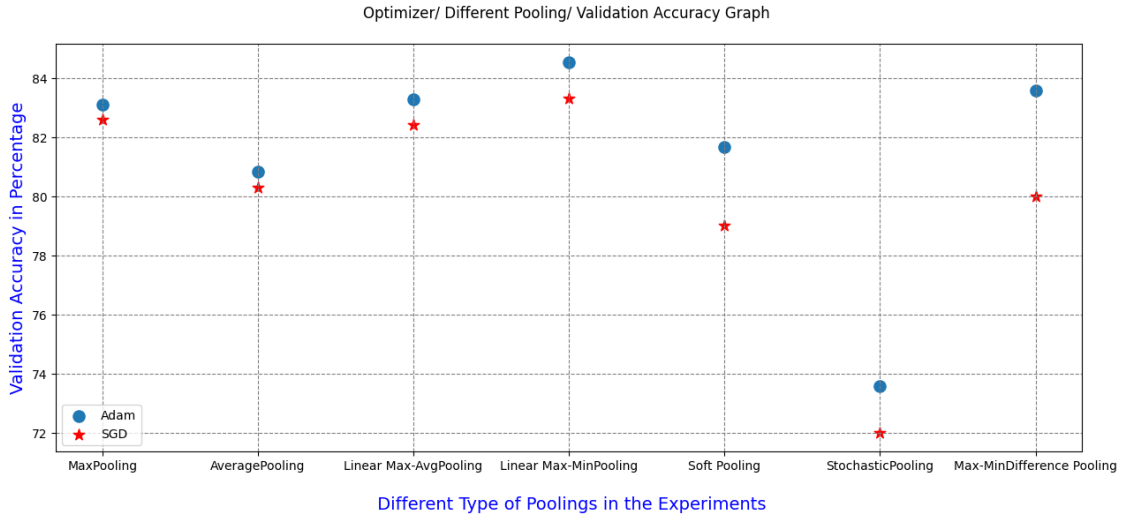


Figure 8: Classification accuracy for first proposed CNN model with different type of pooling layers trained with Adam and SGD. The blue dot corresponds to the Adam optimizer and a red star-shaped character corresponds to SGD optimizer.

The Fig. 8 shows performance of validation accuracy with different pooling methods i.e. max pooling, average pooling, and linear combination of max-average, linear combination of max-min, soft pooling and stochastic pooling with SGD optimizer shown with red asterisks and Adam optimizer shown with blue points. The training accuracy for different pooling methods and optimizers is shown Fig. 9. To address the need for statistical reliability, error bars representing the standard deviation across multiple runs have been added. The graph demonstrates that the proposed Max-Min Difference Pooling method achieves consistently higher accuracy compared to other pooling methods, with minimal variability, as indicated by the narrow error bars. This improvement in visualization provides a clearer understanding of the performance trends and the robustness of the proposed approach.

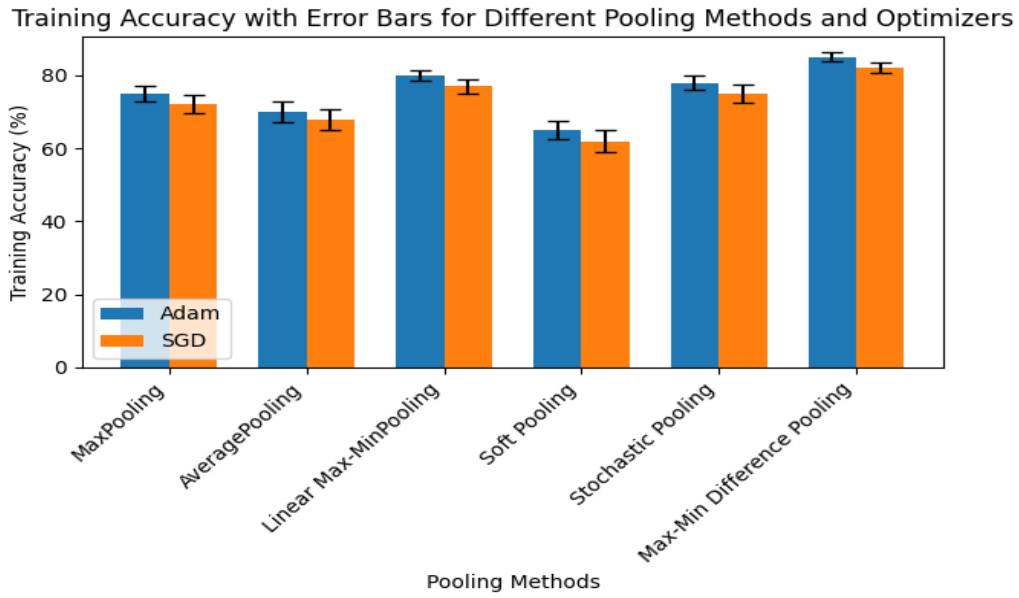


Figure 9: Comparison of training accuracy for proposed CNN model with different pooling layers and optimizers.

The training performance with the entire pooling scheme with both the optimizers is almost same but the difference in performance can be observed for validation accuracy. The linear max-min pooling and newly proposed max-min difference pooling methods are approximately reflecting the same performance on Adam optimizer. The performance analysis on the basis of different parameters i.e., training accuracy, testing accuracy and F1-score for the first CNN architecture with both the optimizers can be shown in Table 4.

Table 4: Training and Testing accuracy and F1- score for first proposed CNN model for both (Adam and SGD) optimizers for the UrbanSound8k dataset.

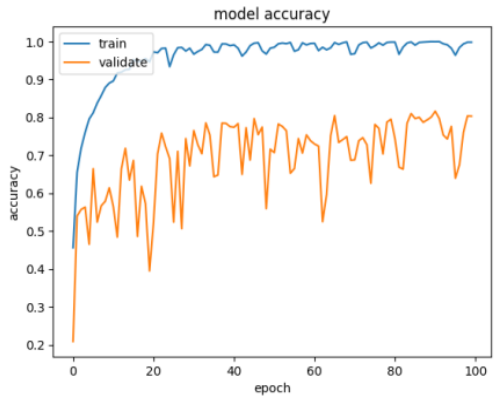
Exp. No.	**Pooling Name**	**Training Accuracy %**		**Testing Accuracy %**		**F1-Score %**	
		Adam	SGD	Adam	SGD	Adam	SGD
1.	Max Pooling	99.42	97.53	83.11	82.59	82	80
2.	Average Pooling	99.75	95.94	80.82	80.30	80	79
3.	Linear Combination Max-Avg Pooling	99.49	97.73	83.28	82.42	75	82
4.	Linear Combination Max-Min Pooling	99.98	97.41	84.54	83.32	75	74
5.	Soft Pooling	99.48	97.12	81.68	80.03	80	79
6.	Stochastic Pooling	99.61	97.47	73.57	72.82	73	72
7.	Max-Min Difference Pooling	99.71	96.19	83.57	80.30	82	80

The performance analysis on the basis of different parameters i.e., training accuracy and testing accuracy for the second CNN architecture with global pooling can be shown in Table 5.

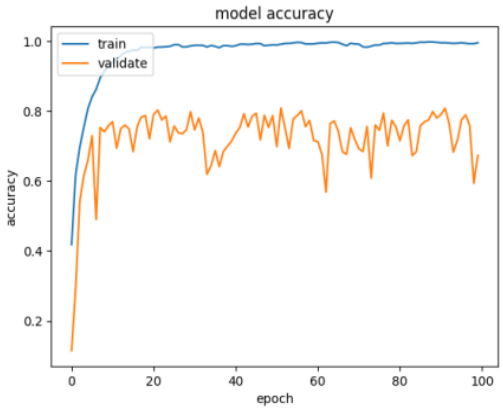
Table 5: Training and testing accuracy for proposed second CNN model for both (Adam and SGD) optimizers for the UrbanSound8k dataset.

Exp. No.	**Pooling Name**	**Training Accuracy %**		**Testing Accuracy %**		**F1-Score %**	
		Adam	SGD	Adam	SGD	Adam	SGD
1.	Max Pooling	97.22	95.94	81.30	80.59	78	77
2.	Average Pooling	97.71	93.75	78.87	77.30	75	78
3.	Linear Comb. Max-Avg Pooling	97.65	95.42	82.05	80.42	79	76
4.	Linear Comb. Max-Min Pooling	97.67	95.41	82.32	81.45	80	76
5.	Soft Pooling	97.23	95.12	79.67	78.43	78	77
6.	Stochastic Pooling	97.33	95.47	71.51	70.78	74	70
7.	Max-Min Difference Pooling	97.12	95.19	81.13	78.76	80	76

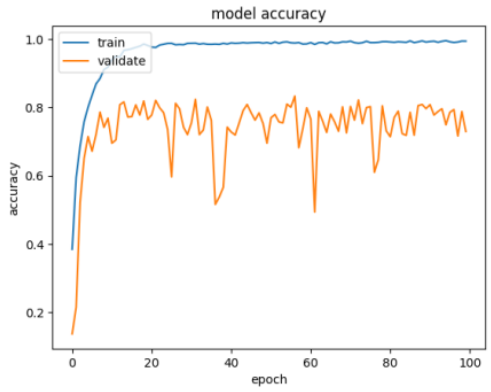
The individual performances of different pooling schemes can be shown in Fig 10. The Fig. 10 is presenting the first CNN model accuracy for training & validation with all the considered pooling schemes. These figures also reflect the more stable behavior of linear combination max-min pooling and max-min difference pooling for both training and validation.



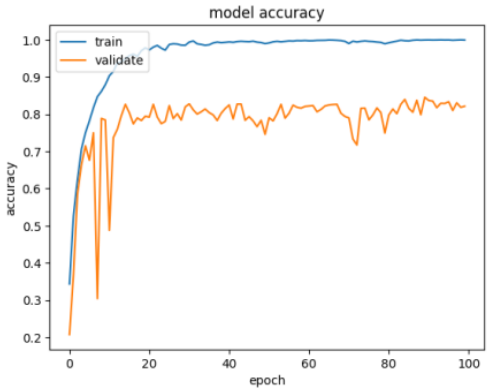
Experiment 1: Proposed CNN model with ax Pooling



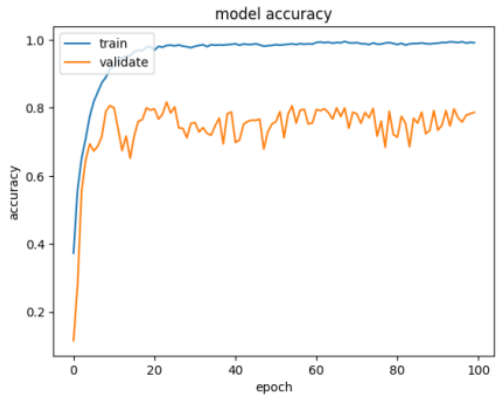
Experiment 2: Proposed CNN model with Average Pooling



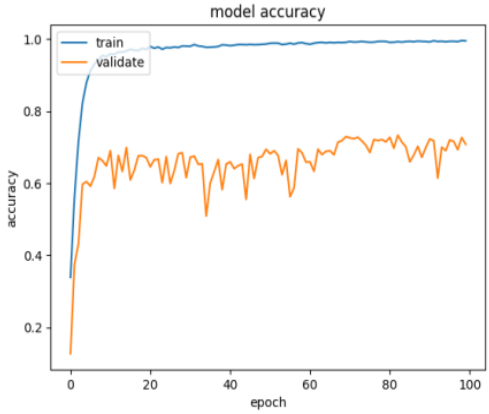
Experiment 3: Proposed CNN model with linear combination max-avg pooling



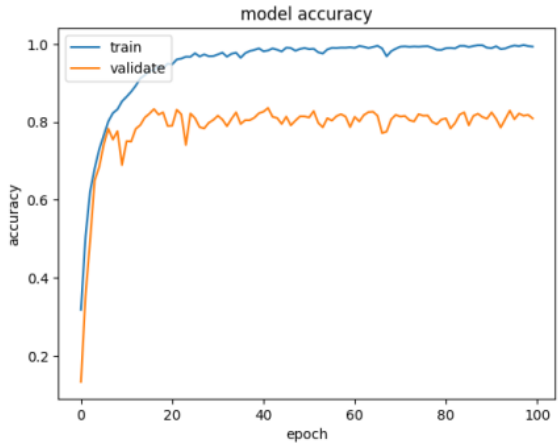
Experiment 4: Proposed CNN model with linear combination max-min pooling



Experiment 5: Proposed CNN model with soft pooling



Experiment 6: Proposed CNN model with stochastic pooling



Experiment 7: Proposed CNN model with max-min combination pooling

Figure 10: Training & validation accuracy for all experiments on proposed first CNN model with different type of pooling evaluated on training and testing data

The confusion matrices of proposed CNN model with linear combination max-min pooling and max-min difference pooling can be seen in Fig. 11 and Fig. 12. Hence, from all the obtained simulated results it can be verified that the performance of CNN model with linear combination max-min pooling and max-min difference pooling is better with Adam optimizer as comparison of other pooling methods. Linear combination max-min pooling exhibits 84.54% accuracy with Adam optimizer and 83.32% accuracy with SGD optimizer. Max-min difference pooling method exhibits 83.57% accuracy in classification with Adam optimizer and 80.30% accuracy with SGD optimizer. The stochastic pooling could not perform well. It has lowest accuracy with respect to other pooling methods for both the optimizers. Linear combination of max-min pooling performed well from the max-min difference pooling for the SGD optimizer. The F1-score of max-min difference pooling is better than all the other pooling methods including linear combination of max-min pooling. Thus, it shows the better performance of max-min difference pooling in confusion matrix. In the present work, we also utilized the confusion matrix to evaluate the performance of CNN architectures. The confusion matrix provides detailed insights into model predictions by displaying the counts of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). This helps identify class imbalances and

misclassifications, offering a clear understanding of the model's performance across different classes. To evaluate the robustness of the proposed convolutional neural network (CNN) architectures, the model was tested on noisy sound data. Confusion matrices were constructed for both the linear combination max-min pooling and the proposed max-min difference pooling, as shown in Fig. 11 and Fig. 12. Then, standard evaluation metrics were employed for statistical analysis to assess the performance of the proposed CNN architectures. Precision, Recall, F1-score, and average accuracy were computed using the following equations (42 to 45):

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+FP+FN+TN}} \tag{42}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \tag{43}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \tag{44}$$

$$\text{F1 score} = \frac{\text{TP}}{\text{TP}+\frac{1}{2}(\text{FP+FN})} \tag{45}$$

The obtained simulated results are interesting and indicate the different trends for classification, in training & validation process with different pooling schemes and different optimizers.

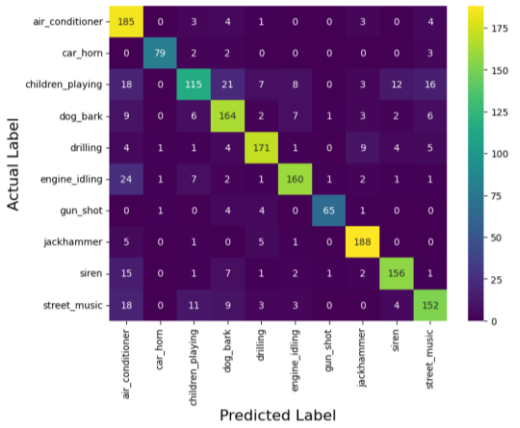


Figure 11: Confusion matrix for proposed CNN with linear combination max-min pooling

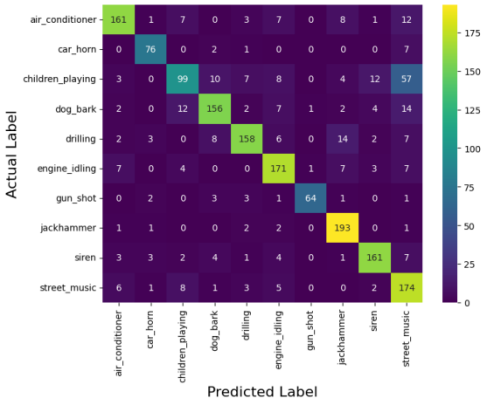
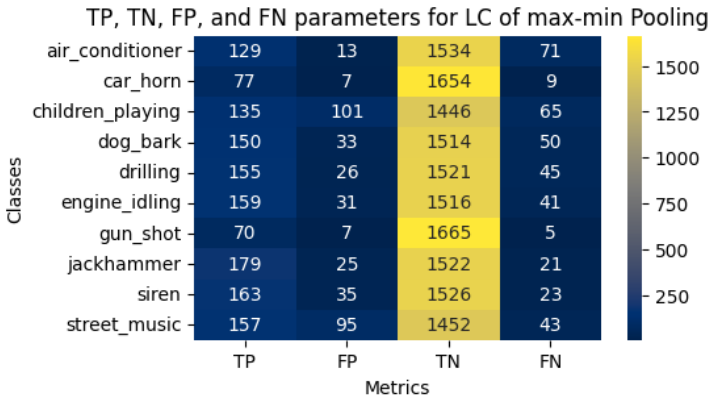


Figure 12: Confusion matrix for proposed CNN with max-min difference pooling

For a better and detailed comparison, we have shown the True Negatives (TN), True Positives (TP), False Positives (FP), and False Negatives (FN) in Fig. 13 for all pooling methods.



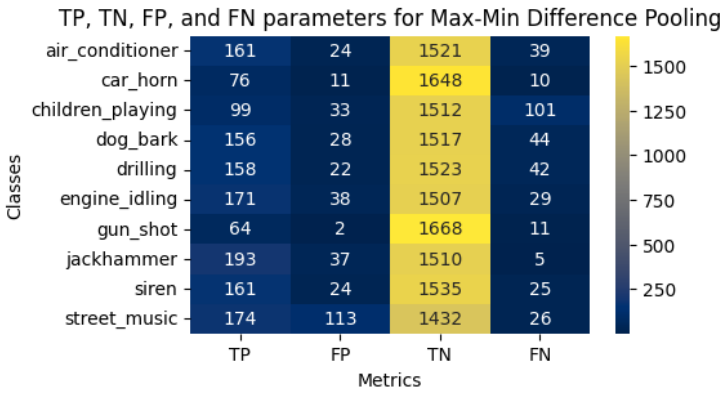


Figure 13: TP, TN, FP, and FN parameters for different pooling methods used in UrbanSound8K audio dataset (Linear Combination of Max-Min Pooling and Max-Min Difference Pooling)

The proposed CNN architectures were trained on the UrbanSound8K dataset using Google Colab with GPU acceleration (NVIDIA Tesla T4/K80/P100). With a batch size of 32 and 100 epochs, the total training process required approximately 8-10 hours which is also includes preprocessing steps for feature extraction. The CNN architecture's computational complexity is O(n), with a total of 68,010 trainable parameters and approximately 1.2 million FLOPs per forward pass. These results demonstrate the efficiency and scalability of the proposed architecture with local pooling for urban sound classification, even on mid-range hardware like Google Colab's GPUs.

Based on our experimental results, it is clearly actionable that for practitioners to guide their choice of pooling methods, CNN architectures, and optimizers. For pooling methods, we recommend using max-min difference pooling for robust feature extraction in noisy environments, while traditional methods like max pooling and average pooling are suitable for computationally constrained tasks. For CNN architectures, the proposed architecture with max-min difference pooling is ideal for small to medium datasets, whereas deeper architecture. In terms of optimizers, Adam is preferred for fast convergence, while SGD with momentum is recommended for fine-tuning. Additionally, practitioners can incorporate dropout and regularization techniques to enhance model performance and prevent overfitting with the choice of Global and local pooling.

5 Conclusion

In this paper, we considered the two convolutional neural network models with different pooling schemes are used for the classification of environmental sound signals. In both the architectures different pooling schemes like max pooling, average pooling, linear combination of max-average, linear combination of max-min, soft pooling and stochastic pooling are used. Along with all these pooling schemes, the paper also explored the new pooling technique named as max-min difference pooling. All these pooling schemes are employed for both the CNN architectures with local, global pooling criteria to analyse the performances. Different simulation results on the various combinations of pooling and optimizers exhibited that the performance of both the CNN architectures with linear combination of max-min pooling is found better for both the optimizer but F1-score of max-min difference pooling for both optimizers is found better. The performance of both the CNNs with all the pooling methods is almost same for training accuracy but the difference occurred in testing accuracy. The performance analysis of CNN is exhibiting

that pooling scheme affects the accuracy in classification, but the choice of appropriate pooling method for the given problem is a critical task. Generally, the stochastic pooling minimizes the problem of max pooling and average pooling but in proposed experimental setup it did not perform well. The accuracy achieved by all pooling methods is found better in comparison of stochastic pooling. The location of pooling layer in CNN also influences the network performance. In our first architecture the local pooling is employed after each convolutional block which includes convolution layer and batch normalization layer. In second architecture the global pooling layer is used after last convolutional block followed by flatten layer and classification layer. In simulation the maximum classification accuracy of 84.54% is obtained. It is also observed that the global pooling scheme reduced the unknown parameters in the network and increases the convergence speed in training but the problem of local minima increases due to unavailability of dense layer. Furthermore, rearrangement and hybrid optimizers may explore to improve the accuracy for sound signal classification. Beside this, the different pooling methods may employ on different convolutional layer of the hybrid CNN architecture to analyse its performance. While the proposed CNN with global pooling shows strong performance, it has some limitations. Global pooling may lead to a loss of fine spatial details, which can impact the CNN architecture's ability to capture subtle features necessary for distinguishing similar classes. Additionally, without dense layers, the model has reduced capacity and flexibility, limiting its effectiveness on tasks that require complex feature interactions or small datasets. Therefore, the proposed pooling scheme, which incorporates a local pooling mechanism and two dense layers, is found to be more effective for the classification of sound data. It provides better accuracy and minimizes the problem of overfitting, which other existing pooling schemes have suffered from. Thus, it is further recommended to use the proposed max-min difference pooling with the convolutional layers, followed by two dense layers, as it provides better performance for the classification of the UrbanSound dataset. Future work could explore hybrid architectures to balance efficiency with enhanced representation learning and we plan to extend our evaluation to additional datasets, such as ESC-50 and AudioSet, and explore real-world sound samples to further validate our findings.

ACKNOWLEDGEMENTS

This research is funded by the Uttar Pradesh government, Lucknow, India in the form of a major research project: 47/2021/606/seventy/4-2020-4(56)/2021.

References

[1] Bansal, A., & Garg, N. K. (2022). Environmental Sound Classification: A descriptive review of the literature. *Intelligent systems with applications, 16*, 200115.

[2] Lu, J., Ma, R., Liu, G., & Qin, Z. (2021, January). Deep convolutional neural network with transfer learning for environmental sound classification. In *2021 International Conference on Computer, Control and Robotics (ICCCR)* (pp. 242-245). IEEE.

[3] Mu, W., Yin, B., Huang, X., Xu, J., & Du, Z. (2021). Environmental sound classification using temporal-frequency attention based convolutional neural network. *Scientific Reports*, *11*(1), 21552.

[4] Bisot, V., Serizel, R., Essid, S., & Richard, G. (2017). Feature learning with matrix factorization applied to acoustic scene classification. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *25*(6), 1216-1229.

[5] Chu, S., Narayanan, S., & Kuo, C. C. J. (2009). Environmental sound recognition with time–frequency audio features. *IEEE Transactions on Audio, Speech, and Language Processing*, *17*(6), 1142-1158.

[6] Tufekci, Z., & Gowdy, J. N. (2000, April). Feature extraction using discrete wavelet transform for speech recognition. In *Proceedings of the IEEE SoutheastCon 2000.'Preparing for The New Millennium'(Cat. No. 00CH37105)* (pp. 116-123). IEEE.

[7] Dhanalakshmi, P., Palanivel, S., & Ramalingam, V. (2009). Classification of audio signals using SVM and RBFNN. *Expert systems with applications*, *36*(3), 6069-6075.

[8] Wang, J. C., Wang, J. F., He, K. W., & Hsu, C. S. (2006, July). Environmental sound classification using hybrid SVM/KNN classifier and MPEG-7 audio low-level descriptor. In *The 2006 IEEE international joint conference on neural network proceedings* (pp. 1731-1735). IEEE.

[9] Bisot, V., Serizel, R., Essid, S., & Richard, G. (2017). Feature learning with matrix factorization applied to acoustic scene classification. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *25*(6), 1216-1229.

[10] Liu, W., Wang, Z., Liu, X., Zeng, N., Liu, Y., & Alsaadi, F. E. (2017). A survey of deep neural network architectures and their applications. *Neurocomputing*, *234*, 11-26.

[11] Lieskovská, E., Jakubec, M., Jarina, R., & Chmulík, M. (2021). A review on speech emotion recognition using deep learning and attention mechanism. *Electronics*, *10*(10), 1163.

[12] Mu, W., Yin, B., Huang, X. *et al.* Environmental sound classification using temporal-frequency attention based convolutional neural network. *Sci Rep* **11**, 21552 (2021).

[13] Kong, Q.; Xu, Y.; Plumbley, M. Sound Event Detection of Weakly Labelled Data With CNN-Transformer and Automatic Threshold Optimization. *IEEE/Acm Trans. Audio Speech Lang. Process.* **2020**, *28*, 2450–2460.

[14] Salamon, J.; Bello, J.P. Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification. *IEEE Signal Process. Lett.* **2017**, *24*, 279–283.

[15] Lee, J., Kim, T., Park, J., & Nam, J. (2017). Raw waveform-based audio classification using sample-level CNN architectures. *arXiv preprint arXiv:1712.00866*.

[16] Abdoli, S., Cardinal, P., & Koerich, A. L. (2019). End-to-end environmental sound classification using a 1D convolutional neural network. *Expert Systems with Applications*, *136*, 252-263.

[17] Hu, C., & Wang, Y. (2020). An efficient convolutional neural network model based on object-level attention mechanism for casting defect detection on radiography images. *IEEE Transactions on Industrial Electronics*, *67*(12), 10922-10930.

[18] Guo, J., Li, C., Sun, Z., Li, J., & Wang, P. (2022). A deep attention model for environmental sound classification from multi-feature data. *Applied Sciences*, *12*(12), 5988.

[19] Li P, Xie J, Wang Q, Zuo W (2017) Is second-order information helpful for large-scale visual recognition*? In: IEEE international conference on computer vision*.

[20] Zeiler, Zeiler, M. D., & Fergus, R. (2013). Stochastic pooling for regularization of deep convolutional neural networks. *arXiv preprint arXiv:1301.3557*.

[21] Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

[22] Wu, H., & Gu, X. (2015). Max-pooling dropout for regularization of convolutional neural networks. *In Neural Information Processing: 22nd International Conference, ICONIP 2015, Istanbul, Turkey*, November 9-12, 2015, Proceedings, Part I 22 (pp. 46-54). Springer International Publishing.

[23] Tong Z, Aihara K, Tanaka G (2016) A hybrid pooling method for convolutional neural networks. *In: International conference on neural information processing*, pp 454–461

[24] Goodfellow, I., Warde-Farley, D., Mirza, M., Courville, A., & Bengio, Y. (2013, May). Maxout networks. *In International conference on machine learning* (pp. 1319-1327). PMLR.

[25] Csurka, G., Dance, C., Fan, L., Willamowski, J., & Bray, C. (2004, May). Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV* (Vol. 1, No. 1-22, pp. 1-2).

[26] He K, Zhang X, Ren S, Sun J (2015) Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Trans Pattern Anal Mach Intell* 37(9):1904–1916

[27] Graham, B. (2014). Fractional max-pooling. *arXiv preprint arXiv:1412.6071*.

[28] Shi, Z., Ye, Y., & Wu, Y. (2016). Rank-based pooling for deep convolutional neural networks. *Neural Networks*, *83*, 21-31.

[29] Gulcehre, C., Cho, K., Pascanu, R., & Bengio, Y. (2014). Learned-norm pooling for deep feedforward and recurrent neural networks. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2014, Nancy, France, September 15-19, 2014. Proceedings, Part I 14* (pp. 530-546). Springer Berlin Heidelberg.

[30] Grauman, K., & Darrell, T. (2005, October). The pyramid match kernel: Discriminative classification with sets of image features. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1* (Vol. 2, pp. 1458-1465).

[31] Tuncer, T., Subasi, A., Ertam, F., & Dogan, S. (2020). A novel spiral pattern and 2D M4 pooling based environmental sound classification method. *Applied Acoustics*, *170*, 107508.

[32] Nirthika, R., Manivannan, S., Ramanan, A., & Wang, R. (2022). Pooling in convolutional neural networks for medical image analysis: a survey and an empirical study. *Neural Computing and Applications*, *34*(7), 5321-5347.

[33] Huang, G., Liu, Z., Weinberger, KQ. (2017). Densely connected convolutional networks. *In: IEEE conference on computer vision and pattern recognition*, pp 2261–2269.

[34] Saha O, Kusupati A, Simhadri HV, Varma M, Jain P (2020) RNNPool: efficient non-linear pooling for ram constrained inference. *arXiv:2002.11921*.

[35] Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

[36] Jarrett, K., Kavukcuoglu, K., Ranzato, M. A., & LeCun, Y. (2009, September). What is the best multi-stage architecture for object recognition?. In *2009 IEEE 12th international conference on computer vision* (pp. 2146-2153). IEEE.

[37] Boureau, Y. L., Ponce, J., & LeCun, Y. (2010). A theoretical analysis of feature pooling in visual recognition. In *Proceedings of the 27th international conference on machine learning (ICML-10)* (pp. 111-118).

[38] Yu, D., Wang, H., Chen, P., & Wei, Z. (2014). Mixed pooling for convolutional neural networks. In *Rough Sets and Knowledge Technology: 9th International Conference, RSKT 2014, Shanghai, China, October 24-26, 2014, Proceedings 9* (pp. 364-375). Springer International Publishing.

[39] Sermanet, P., Chintala, S., & LeCun, Y. (2012, November). Convolutional neural networks applied to house numbers digit classification. In *Proceedings of the 21st international conference on pattern recognition (ICPR2012)* (pp. 3288-3291). IEEE.

[40] Lee CY, Gallagher PW, Tu Z (2016) Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree. *In: Artificial intelligence and statistics*, pp 464–472.

[41] Stergiou A, Poppe R, Kalliatakis G (2021) Refining activation downsampling with SoftPool. *arXiv:2101.00440.*

[42] Song Z, Liu Y, Song R, Chen Z, Yang J, Zhang C, Jiang Q (2018) A sparsity-based stochastic pooling mechanism for deep convolutional neural networks. *Neural Netw* 105:340–345.

[43] Salamon, J., Jacoby, C., & Bello, J. P. (2014, November 3). UrbanSound8K. Zenodo. https://doi.org/10.5281/zenodo.1203745.

[44] Rashmi, P., Singh, M. P., & Prakash, P. (2024). Optimization of Convolutional Neural Network Architectures for High-Accuracy Spoken Digit Classification Using Mel-Frequency Cepstral Coefficients. *Journal of Computational Analysis and Applications*, 33(5).

[45] Singh, M. P., & Rashmi, P. (2024). Convolution Neural Networks of Dynamically Sized Filters with Modified Stochastic Gradient Descent Optimizer for Sound Classification. *Journal of Computer Science*. 20(1):69-87

[46] Gholamalinezhad, H., & Khosravi, H. (2009). Pooling methods in deep neural networks, a review. *arXiv 2020. arXiv preprint arXiv:2009.07485*.

Notes on contributors

Manu Pratap Singh is Professor at the Department of Computer Science, Dr. Bhimrao Ambedkar University, Agra, UP, India. His main teaching and research interests include Artificial Intelligence, Soft Computing, Machine Learning, Deep Learning and Quantum Computing. He has published several research articles in international journals of Computer Science and Physics.

Pratibha Rashmi is Assistant Professor at the Department of Computer Science, Dr. Bhimrao Ambedkar University, Agra, UP, India. Her main teaching and research interests include Machine Learning and Deep Learning. She has published several research articles in international journals of Computer Science.