Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.077087

ARTICLE

Improving Convolutional Neural Network Performance Using Alpha-Based Adaptive Pooling for Image Classification

Nahdi Saubari[1,2,*], **Kunfeng Wang**[1,*], **Rachmat Muwardi**[3,*] **and Andri Pranolo**[4]

[1]College of Information Science and Technology Program of Controlling Science and Engineering, Beijing University of Chemical Technology, Beijing, China
[2]Informatics Department, Universitas Muhammadiyah Banjarmasin, Banjarmasin, Indonesia
[3]Department of Electrical Engineering, Universitas Mercu Buana, Jakarta, Indonesia
[4]Informatics Department, Universitas Ahmad Dahlan, Yogyakarta, Indonesia

*Corresponding Authors: Nahdi Saubari. Email: 2020420001@buct.edu.cn; Kunfeng Wang. Email: wangkf@mail.buct.edu.cn; Rachmat Muwardi. Email: rachmat.muwardi@mercubuana.ac.id

Received: 02 December 2025; Accepted: 10 February 2026; Published: 09 April 2026

ABSTRACT: This study proposes an Adaptive Pooling method based on an alpha (α) parameter to enhance the effectiveness and stability of convolutional neural networks (CNNs) in image classification tasks. Conventional pooling techniques, such as max pooling and average pooling, often exhibit limited adaptability when applied to datasets with heterogeneous distributions and varying levels of complexity. To address this limitation, the proposed approach introduces an α parameter ranging from 0 to 1 that continuously regulates the contribution of maximum-based and average-based pooling operations in a unified and flexible framework. The proposed method is evaluated using two benchmark datasets, MNIST and CIFAR-10, representing grayscale and color image classification scenarios, respectively. Experiments are conducted across three CNN families with different depths LeNet-5, a deeper custom-built CNN, and ResNet-18 to assess robustness under varying representational capacity. Under the best α setting with a 4 × 4 pooling configuration, Adaptive Pooling exhibits architecture-dependent behavior. On LeNet-5, Adaptive Pooling achieves 87.2% on MNIST and 30.1% on CIFAR-10, compared with 97.8% (max/average pooling) on MNIST and 60.1% (max pooling)/53.9% (average pooling) on CIFAR-10. In contrast, on the deeper custom CNN, Adaptive Pooling becomes competitive, reaching 99.7% on MNIST and 86.1% on CIFAR-10, which is comparable to 99.6%–99.7% on MNIST and 84.5%–86.2% on CIFAR-10 achieved by conventional pooling. On ResNet-18, Adaptive Pooling attains 99.1% on MNIST, while CIFAR-10 performance decreases to 37.2% relative to the default global average pooling baseline (99.7% on MNIST and 89.0% on CIFAR-10), suggesting that performance also depends on where the pooling replacement is applied. Overall, these findings indicate that α-controlled Adaptive Pooling provides a lightweight and configurable pooling strategy that can improve stability and achieve competitive accuracy in deeper CNNs, although it should be treated as a complementary mechanism rather than a universal replacement across all architectures.

KEYWORDS: Adaptive pooling; alpha parameter; convolutional neural network (CNN); image classification; MNIST; CIFAR-10; model generalization

1 Introduction

Convolutional Neural Networks (CNNs) have emerged as one of the most effective and popular deep learning architectures in recent decades, particularly for pattern recognition and image classification tasks [1,2]. The capability of CNNs to automatically extract hierarchical features through convolutional

operations provides a major advantage over traditional methods that rely on handcrafted features. Numerous studies have demonstrated the success of CNNs across various image processing domains, including classification, semantic segmentation, and object detection [3].

One of the essential components in CNN architectures is the pooling layer, which serves to reduce the spatial dimensions of feature maps produced by convolutional layers [4,5]. Pooling facilitates the extraction of dominant information while substantially lowering computational costs. The two most widely used pooling types are max pooling and average pooling [6,7]. Max pooling selects the maximum value within a pooling region, while average pooling computes the mean of all values within that region.

Although both pooling methods are widely adopted, each has inherent limitations. Max pooling often discards fine-grained information since it retains only the highest activation values, while average pooling is more sensitive to noise because it aggregates all activations uniformly [8–11]. These drawbacks can hinder CNN performance, especially when processing datasets with high spatial variability and complex feature distributions.

To overcome these limitations, several alternative pooling strategies have been proposed to enhance feature representation in CNNs. A recent approach introduced in [11], called T-Max-Avg pooling, employs a threshold-based mechanism to dynamically switch between max and average pooling. This method combines the concept of Top-K selection [12] with a threshold parameter (T) to extract the most representative features within each pooling region: if the maximum value in a window exceeds the threshold, max pooling is applied; otherwise, average pooling is used.

Pooling remains a fundamental operation in convolutional neural networks because it reduces spatial redundancy, enhances translational robustness, suppresses noise, and stabilizes feature representations across network layers. However, conventional max pooling and average pooling operate with fixed aggregation behavior, making them less adaptive to variations in feature distribution and network depth [13].

In contrast to discrete and non-trainable threshold-based pooling strategies, this study proposes an Adaptive Pooling method based on linear interpolation controlled by a single α parameter. The method combines max pooling and average pooling in a continuous manner, where α regulates the balance between dominant feature selection and global contextual aggregation. Unlike fixed threshold mechanisms, this approach enables smooth adjustment of pooling behavior according to feature characteristics while maintaining a simple and computationally efficient design.

Beyond its mathematical formulation, the proposed Adaptive Pooling mechanism provides an interpretable control parameter that allows direct examination of how pooling behavior influences model stability and generalization. The experimental findings further reveal that α behaves as an architecture-dependent pooling controller: in shallow networks its effect is limited, whereas in deeper CNNs with richer hierarchical feature representations it becomes increasingly beneficial. This property highlights the flexibility of the method and its suitability for modern deep learning architectures.

In this study, several fixed α values are evaluated across different pooling sizes and datasets to analyze their effect on classification accuracy and training dynamics. Model performance is assessed using accuracy, precision, recall, and F1-score, while feature visualization and ROC analysis are used to examine the impact of Adaptive Pooling on spatial representation learning and discriminative capability. Overall, the proposed method introduces a computationally efficient and flexible pooling strategy that enhances feature representation without increasing model complexity.

Main contributions of this study are summarized as follows:

1. A lightweight adaptive pooling strategy that combines max pooling and average pooling using a single alpha parameter.

This study introduces a unified pooling mechanism that continuously interpolates between max pooling and average pooling through a single α parameter, providing controllable feature aggregation while preserving computational simplicity and avoiding additional trainable parameters.

2. The proposed method introduces no additional parameters and does not increase training cost compared to conventional pooling.
   The proposed method is formulated such that it can be integrated into both shallow and deep convolutional networks without modifying the backbone structure, enabling fair architectural comparison and facilitating cross-model validation.
3. The method is evaluated on both the MNIST and the CIFAR datasets to demonstrate robustness and generalization capability.
   The influence of α is examined under multiple pooling window sizes and across datasets of different visual characteristics, allowing a deeper understanding of how α regulates the balance between dominant feature selection and contextual smoothing.
4. The influence of the alpha parameter on accuracy and training stability is systematically investigated.
   The study provides a principled evaluation setup that compares Adaptive Pooling with conventional pooling strategies under identical training configurations, enabling objective analysis of how pooling behavior interacts with network depth and feature hierarchy.

The remainder of this paper is organized as follows. Section 2 reviews related work on pooling techniques. Section 3 describes the proposed Adaptive Pooling method. Section 4 reports the experimental results and analysis. Section 5 presents additional evaluation on a deeper CNN architecture. Finally, Section 6 concludes the paper.

2 Related Work

Pooling is one of the key components in Convolutional Neural Network (CNN) architectures, designed to reduce spatial dimensionality, suppress overfitting, and lower computational costs. Conventional pooling methods, such as max pooling and average pooling, have been widely adopted due to their simplicity and effectiveness. However, both approaches exhibit limitations in capturing complex spatial structures and are highly sensitive to noise and the loss of fine-grained details [14,15].

Recent studies have revisited the role of pooling by introducing smoother and information-preserving downsampling operators such as SoftPool and exponential adaptive pooling to mitigate information loss during feature map reduction [14,16,17]. This issue remains particularly relevant in modern CNN architectures, as spatial downsampling is still a core design element for computational efficiency and hierarchical representation learning. Consequently, the choice of pooling or downsampling strategy can substantially influence feature quality and model generalization performance [18–21].

To address these shortcomings, several studies have proposed alternative pooling strategies that aim to be more adaptive and computationally efficient. One notable example is Universal Pooling (UP), developed by [22]. UP reformulates pooling as a channel-wise attention mechanism that is both spatially and locally adaptive. Instead of using a fixed pooling function, UP learns pooling weights in an end-to-end manner alongside the main network. As a result, it can produce pooling functions that vary according to data characteristics. While UP effectively improves accuracy, it incurs significant computational overhead due to the additional trainable pooling weight module. Other recent approaches also pursue adaptivity by estimating importance attention within local regions before aggregation, improving feature selection under challenging variations [23,24].

A different line of research emphasizes computational efficiency, exemplified by Wavelet Pooling introduced in [15]. This approach employs a two-level wavelet transform for feature subsampling. Unlike

neighborhood-based pooling operations, Wavelet Pooling better preserves spatial structure and mitigates artifacts such as blurring and aliasing. Nevertheless, the two-level wavelet transformation and reconstruction process increase the overall computational complexity compared to conventional max or average pooling.

Expanding on this concept, Multiple Wavelet Pooling (MWP) was proposed by [25] explores combinations of multiple wavelet bases such as Haar, Daubechies, and Coiflet to construct more flexible pooling functions. In this method, two different wavelet-transformed outputs are fused to form a new feature representation from the low-frequency sub-band (LL). Experiments on datasets such as MNIST, CIFAR-10, and SVHN demonstrated that MWP outperforms conventional pooling in several cases, particularly when dropout regularization is applied. However, the method introduces additional architectural complexity, which can hinder scalability.

In addition to pooling-based strategies, image enhancement through pixel-level fusion has also been explored as a means to improve recognition performance. A recent study proposed an opposite-frequency fusion framework based on discrete wavelet transform combined with Gaussian and Difference of Gaussian filtering, in which low-frequency components were fused using DoG and high-frequency components using Gaussian filtering. The reconstructed images, obtained through inverse wavelet transform, were shown to significantly enhance face recognition accuracy across multiple benchmark datasets and challenging conditions such as illumination variation, head pose, occlusion, and image degradation [26,27].

Furthermore, multiscale analysis and curvature-based features have also been investigated to strengthen structural representation in visual recognition tasks. A previous study combined multiscale representations with curvature-based descriptors to enhance discrimination capability in partial face recognition, demonstrating that structural information extracted from different scales can improve robustness against facial variations [28]. Although this approach improves recognition accuracy, it relies heavily on handcrafted feature extraction rather than learned representations.

Another noteworthy approach is T-Max-Avg Pooling, proposed by [11] dynamically selects between max and average pooling based on a predefined threshold parameter (T). By filtering pixel activations according to the highest values and considering the top-K most significant features, the method captures more representative spatial information. Experiments on CIFAR-10, CIFAR-100, and MNIST datasets have shown that T-Max-Avg Pooling surpasses traditional pooling methods in classification accuracy [29]. However, its adaptability is limited by reliance on a discrete threshold, making it less flexible for continuous optimization or end-to-end training.

Overall, existing pooling methods involve a trade-off between adaptivity, computational efficiency, and implementation simplicity. Many adaptive approaches improve accuracy by enhancing feature selection but often increase model complexity or training cost.

To address this issue, this study proposes a lightweight Adaptive Pooling method controlled by a single alpha parameter to achieve flexible pooling behavior while using a single predefined alpha coefficient and avoiding additional trainable pooling weights or auxiliary modules. The proposed design enables efficient integration into different CNN architectures and datasets.

3 Method

This section presents the proposed Adaptive Pooling methodology, including the CNN architecture, formulation of the pooling mechanism, and the experimental setup used for performance evaluation.

3.1 General Architecture of Convolutional Neural Networks

A CNN typically consists of three to five main layers, including convolutional layers, pooling layers, and fully connected layers [30,31]. The convolutional layers are responsible for extracting spatial features from the input image, while the pooling layers reduce the spatial dimensions of the feature maps to enhance computational efficiency and strengthen translational invariance [32]. The general architecture of a CNN is illustrated in Fig. 1.

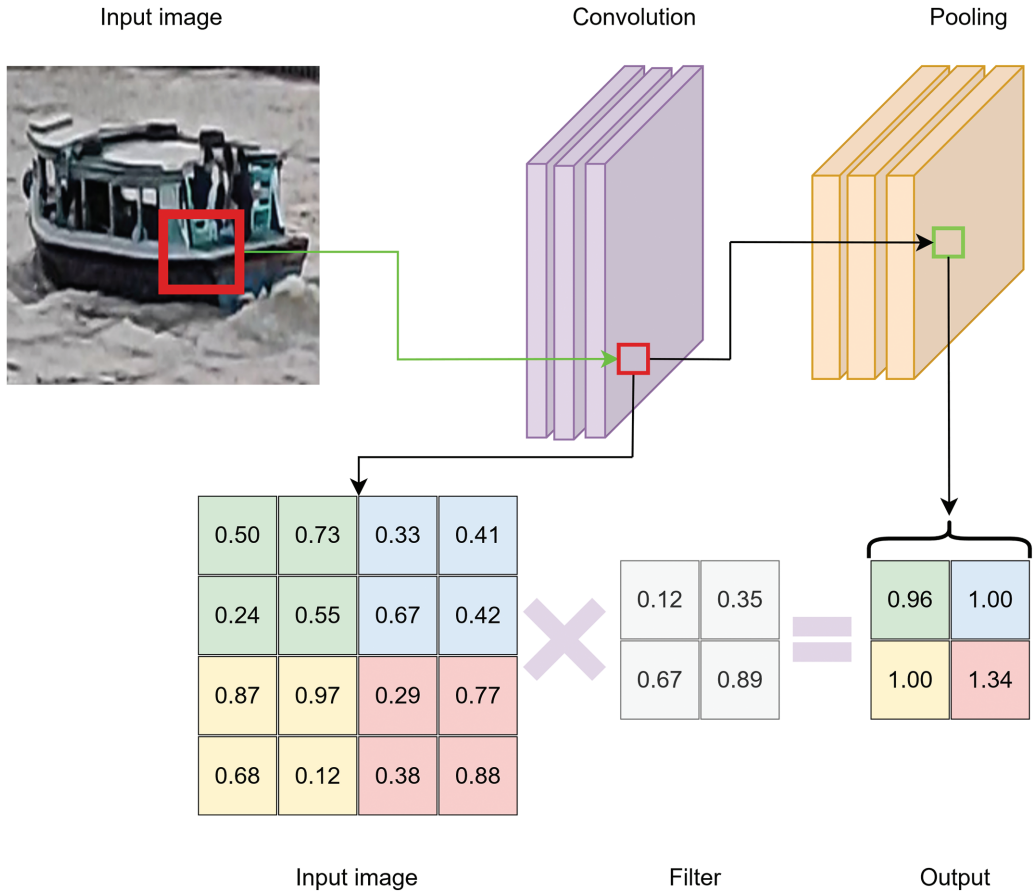


Figure 1: The general architecture of a CNN.

3.2 Max-Pooling

Max pooling is a widely used operation in CNN feature extraction that reduces spatial dimensions by selecting the maximum value within each local region. This process preserves the most salient activation in each region, allowing the network to retain dominant structural patterns such as edges and textures while lowering computational cost [13,33].

However, because only the highest activation is retained, max pooling may discard fine-grained information and degrade feature descriptiveness. The mathematical formulation of max pooling is given in Eq. (1).

$$O_{\max(x)} = max\left(x_1, x_2, x_3 \ldots x_n\right) \tag{1}$$

here, $O_{max(x)}$ denotes the output of the max pooling operation, and x_1 to x_n represent the pixel values within a local pooling region.

Fig. 2 illustrates the max pooling process using a 2 × 2 filter on a 4 × 4 input image, resulting in the corresponding pooled feature map.

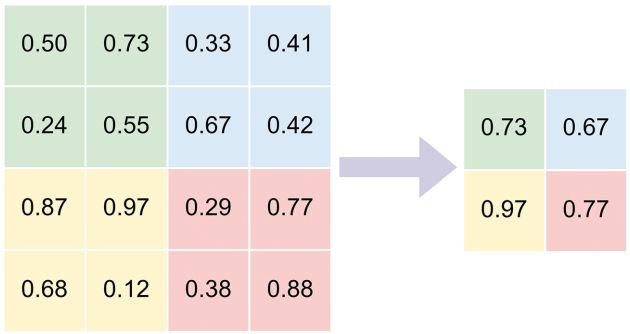


Figure 2: Max-pooling.

3.3 Average Pooling

Average pooling reduces spatial dimension by computing the mean value within each local region of an image [34,35]. This operation provides smoother feature maps and improves robustness to noise.

However, averaging may blur important local details and weaken feature discriminability. The mathematical formulation of average pooling is given in Eq. (2).

$$O = \frac{1}{n}\sum_{i-1}^{n} x_i \tag{2}$$

where O represents the output of the average pooling operation, x_i denotes the pixel values within a pooling region, and n is the number of elements in that region.

Fig. 3 illustrates the average pooling process using a 2 × 2 filter applied to a 4 × 4 input image.

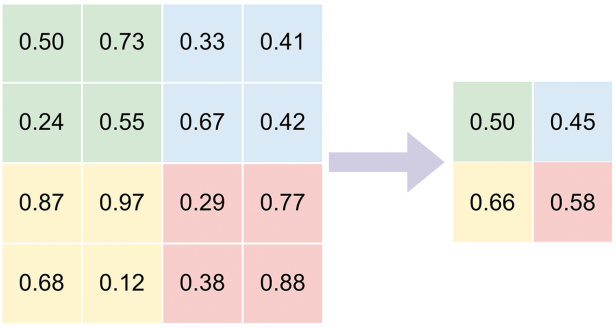


Figure 3: Average pooling.

3.4 Purpose Pooling Method

Unlike conventional static pooling methods, this study introduces a new pooling formulation referred to as Adaptive Pooling, which is controlled by a scalar alpha parameter ranging from 0 to 1. The proposed formulation balances the contribution between max pooling and average pooling in a single continuous expression.

The output of the pooling operation $O_{(x)}$ for a local region $x = \{x_1, x_2, x_3, \ldots, x_n\}$ is defined as shown in Eq. (3):

$$O_{(x)} = \alpha \cdot MaxPool\,(x) + (1 - \alpha) \cdot AvgPool\,(x) \tag{3}$$

The alpha parameter is predefined before training. When $\alpha = 1$, the operation becomes max pooling, while $\alpha = 0$ corresponds to average pooling. For values between 0 and 1, the method integrates both strategies, maintaining sensitivity to salient features while suppressing noise and outliers.

In this study, alpha is varied from 0.1 to 0.95 to evaluate model performance across different datasets. Fig. 4 illustrates the Adaptive Pooling process using a 2 × 2 filter on a 4 × 4 input image.

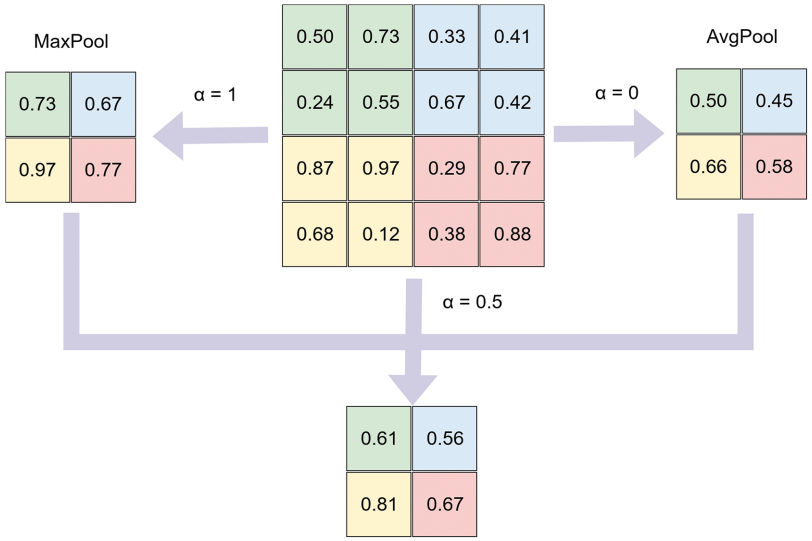


Figure 4: Adaptive pooling with α parameter.

3.5 Created CNN Network

To evaluate the proposed Adaptive Pooling method, this study adopts the LeNet-5 architecture introduced by LeCun [36] as the baseline model due to its simplicity, efficiency, and proven performance in image classification.

LeNet-5 consists of three convolutional layers, two pooling layers, and two fully connected layers. This architecture is used to compare Adaptive Pooling with conventional pooling methods, including Avg-TopK and T-Max-Avg. The network structure is detailed in Table 1, and the schematic illustration is shown in Fig. 5.

Table 1: LeNet-5 network architecture.

Layer	Layer Type	Size of Feature Map	Kernel Size	Param	Activation Function
Input Layer	Image	32 × 32 × 3	–	–	–
Conv1 layer	Convolution	32 × 32 × 6	5 × 5	456	Tanh
Pool1 layer	CustomPool1	16 × 16 × 6	2 × 2	0	–
Conv2 layer	Convolution	16 × 16 × 16	5 × 5	2416	Tanh
Pool2 layer	CustomPool2	8 × 8 × 16	2 × 2	0	–

(Continued)

Table 1 (continued)

Layer	Layer Type	Size of Feature Map	Kernel Size	Param	Activation Function
Conv3 layer	Convolution	8 × 8 × 20	5 × 5	48,120	Tanh
FC1 layer	Full connected	84	–	645,204	Tanh
FC2 layer	Full connected	10	–	850	Softmax

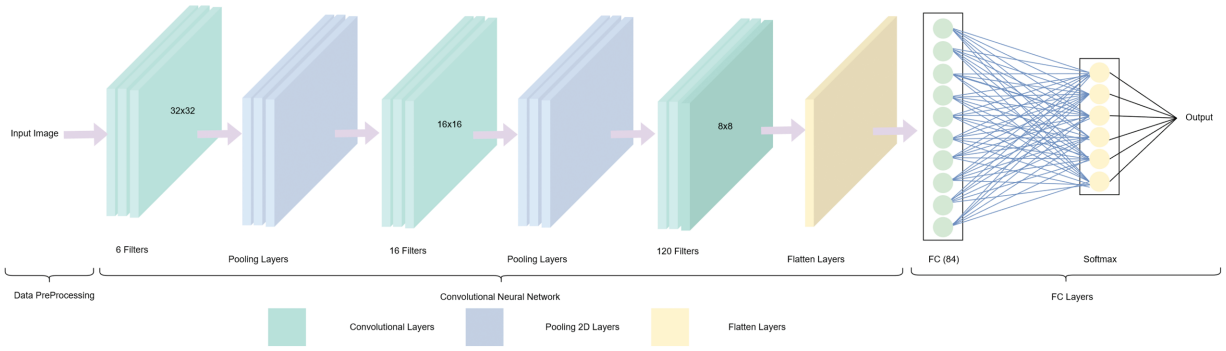


Figure 5: LeNet-5 model architecture.

All experiments were conducted on MNIST and CIFAR-10 under identical training conditions for both the proposed and reference methods to ensure fair evaluation.

The LeNet-5 architecture is summarized in Table 1 follows a classical design with alternating convolution and pooling layers to reduce spatial resolution while progressively expanding feature representation. Each convolution layer uses 5 × 5 kernels with Tanh activation, while the pooling layers, including the proposed Adaptive Pooling and baseline methods, perform spatial downsampling and improve translation invariance.

The resulting feature maps are flattened and processed by two fully connected layers before classification by the Softmax layer. This lightweight and well-controlled architecture allows reliable evaluation of the specific impact of the proposed pooling method.

4 Experimental Result

This section reports the experimental results and performance analysis of the proposed Adaptive Pooling method, including quantitative evaluation, training behavior, and comparison across architectures and pooling strategies.

4.1 Dataset

In this section, we describe the datasets used to evaluate and compare the performance of the proposed Adaptive Pooling method with conventional pooling approaches. Two public benchmark datasets, MNIST and CIFAR-10, were employed in the experiments.

4.2 MNIST

MNIST is a widely used benchmark dataset for handwritten digit classification in computer vision [37]. It contains 70,000 grayscale images of size 28 × 28 pixels across 10 classes, with 60,000 samples for training and 10,000 for testing.

MNIST is commonly used for preliminary evaluation due to its simple structure and low resolution. In this study, it is adopted to examine the effectiveness of the proposed Adaptive Pooling method on datasets with simple feature distributions.

4.3 CIFAR-10

CIFAR-10 is a widely used benchmark dataset for image classification in computer vision [38]. It contains 60,000 RGB images of size 32 × 32 pixels across 10 balanced classes, with 50,000 training samples and 10,000 testing samples.

CIFAR-10 is commonly used to evaluate model performance and generalization ability on complex visual data. In this study, CIFAR-10 is employed to assess the effectiveness of the proposed Adaptive Pooling method compared with conventional pooling techniques [39,40].

4.4 Network Parameters

The models were trained using the Adam optimizer with categorical cross-entropy loss, and all input images were normalized to the range 0–1. Training was performed for up to 50 epochs with a batch size of 16. To ensure a fair and controlled evaluation of the proposed pooling operator, the network architecture and all training settings were kept identical across experiments; only the pooling layer was varied (Max Pooling, Average Pooling, and the proposed Adaptive Pooling). No transfer-learning fine-tuning was performed, as the goal of this study is to isolate the impact of pooling-layer replacement rather than to redesign or tune the overall architecture. In the baseline models, the final pooling layer used conventional pooling (Max or Average), while in the proposed configuration it was replaced by the Adaptive Pooling layer. EarlyStopping and ReduceLROnPlateau were applied to improve convergence, the best-performing model was saved using ModelCheckpoint, and training time was recorded to assess computational efficiency.

4.5 Experimental Result

Two benchmark datasets, MNIST and CIFAR-10, were used to evaluate the proposed method, representing grayscale and color image classification tasks, respectively. The experiments were conducted using the LeNet-5 architecture with pooling sizes ranging from 2 to 4 to analyze the effect of pooling scale on classification performance.

All experiments were implemented in Jupyter Notebook using an NVIDIA GeForce RTX 3060 GPU to accelerate training. The proposed method and baseline approaches (max pooling, average pooling, Avg-TopK, and T-Max-Avg) were evaluated under identical model configurations and training settings to ensure a fair comparison.

Each experiment was repeated six times, and the reported results represent the average accuracy across all runs. The classification performance for pooling size 2 on MNIST and CIFAR-10 is summarized in Tables 2 and 3.

Table 2: Experimental results with Pool size = 2 on the CIFAR-10 dataset.

α Value	Exp-1 (%)	Exp-2 (%)	Exp-3 (%)	Exp-4 (%)	Exp-5 (%)	Exp-6 (%)	Average (%)
0.1	25.5	25.3	25.8	23.7	24.6	24.4	24.9
0.15	24.3	25.5	26.7	23.3	26.4	24.0	25.0
0.2	23.3	21.0	23.0	25.3	25.2	26.0	24.0
0.25	24.8	24.9	24.4	26.1	24.7	20.7	24.3
0.3	25.0	24.8	24.1	24.7	26.4	26.0	25.2
0.35	24.9	23.7	24.5	20.6	23.3	24.6	23.6
0.4	26.7	25.5	20.8	27.0	10.0	24.3	22.4
0.45	23.5	21.7	21.1	23.1	22.6	21.2	22.2
0.5	22.4	21.0	21.4	21.4	23.7	21.8	22.0
0.55	24.5	21.8	24.5	22.6	21.9	21.2	22.7
0.6	21.9	24.7	10.0	23.3	23.3	23.4	21.1
0.65	20.6	21.8	22.1	23.2	20.6	24.4	22.1
0.7	25.1	24.8	24.6	21.4	20.8	21.5	23.0
0.75	21.2	22.7	24.2	24.5	21.0	10.0	20.6
0.8	22.9	23.1	21.8	23.4	20.9	20.4	22.1
0.85	20.7	24.7	10.0	21.8	22.1	23.8	20.5
0.9	23.2	22.4	10.0	17.6	22.6	20.9	19.4
0.95	10.0	10.0	12.9	22.7	10.0	23.0	14.7
Max	49	49.9	49	50.2	50.69	50.16	50
Avg	57	56.7	57.3	57	60.37	60.02	58.1

Table 3: Experimental results with Pool size = 2 on the MNIST dataset.

α Value	Exp-1 (%)	Exp-2 (%)	Exp-3 (%)	Exp-4 (%)	Exp-5 (%)	Exp-6 (%)	Average (%)
0.1	66.7	69.0	61.2	63.3	58.6	64.7	63.9
0.15	60.1	66.9	63.0	67.9	57.7	65.9	63.6
0.2	62.5	62.7	63.9	58.9	61.6	59.7	61.5
0.25	62.9	62.1	62.1	60.9	64.0	64.3	62.7
0.3	62.0	65.0	61.6	57.9	62.3	64.1	62.1
0.35	64.1	64.9	63.8	59.2	63.0	64.2	63.2
0.4	62.8	63.0	62.6	59.2	63.1	63.4	62.4
0.45	59.7	59.7	62.8	65.3	64.0	63.8	62.5
0.5	63.4	62.0	62.6	60.7	59.5	63.0	61.8
0.55	64.9	64.8	63.9	60.5	64.4	62.6	63.5
0.6	65.1	64.4	63.8	62.5	60.4	64.2	63.4
0.65	65.0	65.4	63.8	65.1	62.0	65.0	64.4
0.7	64.2	61.4	61.8	63.1	64.1	65.1	63.3
0.75	62.2	61.9	56.9	61.9	65.4	65.9	62.4
0.8	63.1	63.0	66.5	65.5	60.1	65.6	64.0
0.85	62.5	64.9	64.3	65.3	61.3	65.1	63.9
0.9	65.4	66.5	65.3	60.8	66.5	64.7	64.9

(Continued)

Table 3 (continued)

α Value	Exp-1 (%)	Exp-2 (%)	Exp-3 (%)	Exp-4 (%)	Exp-5 (%)	Exp-6 (%)	Average (%)
0.95	63.5	64.8	65.7	66.4	62.5	64.5	64.6
Max	98.63	98.71	98.51	98.59	98.61	98.67	98.62
Avg	98.38	98.52	98.3	97.96	98.36	98.75	98.38

For pooling size 2, traditional pooling methods consistently outperformed Adaptive Pooling on both datasets. On CIFAR-10, average pooling achieved the highest accuracy (58.1%), followed by max pooling (50%), while Adaptive Pooling achieved only 25.2% at its best alpha setting. Similarly, on MNIST, max pooling (98.62%) and average pooling (98.38%) significantly surpassed Adaptive Pooling, which reached a maximum accuracy of 64.6%.

Overall, these results indicate that with small pooling sizes, max pooling and average pooling remain more effective and stable than the proposed Adaptive Pooling method.

The experimental results for both traditional and proposed pooling methods using a pooling size of 3 are presented in Tables 4 and 5.

Table 4: Experimental results with Pool size = 3 on the CIFAR-10 dataset.

α Value	Exp-1 (%)	Exp-2 (%)	Exp-3 (%)	Exp-4 (%)	Exp-5 (%)	Exp-6 (%)	Average (%)
0.1	28.1	31.1	29.3	30.8	26.4	30.7	26.5
0.15	29.9	30.7	28.8	29.7	28.1	29.3	26.3
0.2	28	31.3	28.7	29.1	26.9	28.9	26.1
0.25	29	28.5	27.8	29.9	30.9	28.8	26
0.3	30.7	30.7	30.1	31.1	29.6	26.8	25.7
0.35	28.1	28.1	26.8	25.8	29.5	27.8	25.4
0.4	31.4	28.8	27.3	26.7	27.9	29.6	25.2
0.45	29	28.4	30.8	26.6	27.2	30.3	24.9
0.5	29.6	29.2	26.5	30.1	26.4	29.2	24.5
0.55	31.5	30.9	26.6	30.3	30.9	28.2	24.1
0.6	27.5	29.6	26.5	26.5	29.8	29.9	23.4
0.65	31.1	27.6	29.6	26.0	26.1	30.8	22.7
0.7	26.5	25.2	30.8	26.1	30.1	29.9	21.7
0.75	25.1	29.4	29.5	26.7	27.7	30.0	20.4
0.8	10	10.0	25.0	10.0	29.7	10	18.5
0.85	26.5	28.3	24.0	27.9	26.0	29.8	19.4
0.9	10	23.4	10	24.9	10.0	24.4	15.6
0.95	10.0	10.0	10.0	19.3	25.6	10.0	14.1
Max	61	61.4	59.78	59.5	60.7	60.7	60.4
Avg	55	54.6	54.31	55.7	55	53.84	55

Table 5: Experimental results with Pool size = 3 on the MNIST dataset.

α Value	Exp-1 (%)	Exp-2 (%)	Exp-3 (%)	Exp-4 (%)	Exp-5 (%)	Exp-6 (%)	Average (%)
0.1	78.6	81.9	80.1	79.6	78.3	80.2	79.8
0.15	72.0	77.8	78.3	80.3	80.1	77.4	77.6
0.2	79.8	80.6	78.0	78.3	78.7	78.5	79.0
0.25	78.0	77.9	77.6	78.6	78.8	78.8	78.3
0.3	78.4	79.3	79.1	78.8	78.2	78.0	78.6
0.35	78.5	78.2	80.4	79.1	78.3	79.4	79.0
0.4	79.4	78.6	79.8	77.7	79.9	79.3	79.1
0.45	77.1	79.5	69.0	77.2	78.7	78.0	76.6
0.5	79.5	77.7	78.2	78.3	77.4	78.9	78.3
0.55	78.9	76.9	78.6	78.2	78.1	78.4	78.2
0.6	78.1	79.5	79.1	78.3	67.5	79.6	77.0
0.65	79.2	78.0	79.4	78.8	72.8	78.1	77.7
0.7	78.5	78.1	78.7	79.4	79.2	78.7	78.8
0.75	80.2	79.6	79.3	79.6	79.5	70.3	78.1
0.8	78.6	71.9	72.2	76.9	77.7	71.1	74.7
0.85	72.3	79.5	72.1	79.0	78.8	80.1	77.0
0.9	78.7	70.7	77.8	79.0	79.7	79.6	77.6
0.95	78.4	78.2	79.1	78.1	72.5	78.7	77.5
Max	98.7	98.47	98.46	98.66	98.73	98.74	98.63
Avg	98.72	98.4	98.36	98.74	98.66	98.52	98.57

For pooling size 3, Adaptive Pooling performed substantially worse than traditional pooling methods on both datasets. On CIFAR-10, the best accuracy reached only 26.5% at alpha = 0.1, which was more than 30% lower than max pooling (60.4%) and average pooling (55%). On MNIST, Adaptive Pooling achieved its highest accuracy of 78.3% at alpha = 0.25, but still lagged behind max pooling (98.63%) and average pooling (98.57%) by approximately 20%.

These findings indicate that with medium pooling size, the proposed Adaptive Pooling method remains highly sensitive to data characteristics and alpha selection, and is not yet competitive with conventional pooling. Nevertheless, these results highlight the method's limitations and motivate further improvements, such as integrating more complex architectures or incorporating attention mechanisms.

The classification accuracies obtained using a pooling size of 4 for both traditional and proposed pooling methods on the datasets are presented in Tables 6 and 7.

Table 6: Experimental results with Pool size = 4 on CIFAR-10 dataset.

α Value	Exp-1 (%)	Exp-2 (%)	Exp-3 (%)	Exp-4 (%)	Exp-5 (%)	Exp-6 (%)	Average (%)
0.1	30.8	29.7	30.4	27.2	28.9	28.1	29.2
0.15	30.2	30.5	30.6	26.7	31.2	31.2	30.1
0.2	31.2	29.9	27.4	27.6	28.5	28.6	28.9
0.25	32.1	31.1	28.9	26.0	29.1	30.0	29.5
0.3	28.6	28.4	28.3	27.8	30.4	28.3	28.6

(Continued)

Table 6 (continued)

α Value	Exp-1 (%)	Exp-2 (%)	Exp-3 (%)	Exp-4 (%)	Exp-5 (%)	Exp-6 (%)	Average (%)
0.35	31.3	26.9	29.4	29.5	30.0	28.3	29.2
0.4	29.1	29.6	27.2	27.2	28.9	27.2	28.2
0.45	28.3	30.4	27.5	26.6	27.6	27.5	28.0
0.5	30.0	26.9	27.8	26.8	28.2	30.5	28.4
0.55	29.2	31.1	29.1	29.2	27.2	29.1	29.2
0.6	29.6	27.2	27.0	26.8	27.3	28.1	27.7
0.65	30.3	26.8	26.6	26.7	28.8	26.1	27.5
0.7	10.0	29.6	27.9	26.6	26.9	27.6	24.8
0.75	10.0	24.0	19.6	10.0	28.0	27.8	19.9
0.8	20.9	19.0	25.9	10.0	10.0	14.0	16.6
0.85	26.7	27.8	25.4	10.0	10.2	24.1	20.7
0.9	20.7	27.1	17.4	21.1	9.8	10.0	17.7
0.95	10.0	9.9	10.0	10.0	10.0	10.0	10.0
Max	–	–	–	–	–	–	60.1
Avg	–	–	–	–	–	–	53.9

Table 7: Experimental results with Pool size = 4 on MNIST dataset.

α Value	Exp-1 (%)	Exp-2 (%)	Exp-3 (%)	Exp-4 (%)	Exp-5 (%)	Exp-6 (%)	Average (%)
0.1	84.5	84.8	88.2	87.9	87.2	86.7	86.6
0.15	86.8	84.5	88.0	85.6	85.8	85.3	86.0
0.2	87.9	87.9	84.7	85.5	85.8	86.2	86.3
0.25	85.9	85.4	86.9	84.8	87.1	88.1	86.4
0.3	86.4	85.4	88.1	86.2	88.1	86.4	86.8
0.35	87.5	87.9	86.5	88.0	86.5	85.4	87.0
0.4	86.5	88.0	86.6	87.3	87.9	85.6	87.0
0.45	86.5	85.5	87.7	87.1	85.9	83.6	86.0
0.5	85.1	86.9	87.9	88.7	85.9	88.7	87.2
0.55	87.3	87.6	87.1	88.0	85.6	87.0	87.1
0.6	88.0	86.1	86.6	87.4	87.2	87.0	87.1
0.65	87.7	84.6	88.1	86.7	85.7	88.0	86.8
0.7	86.3	86.1	87.3	87.1	87.8	87.3	87.0
0.75	87.8	87.2	86.5	85.1	86.1	87.4	86.7
0.8	85.4	86.2	82.6	87.2	87.3	86.9	85.9
0.85	84.9	88.5	87.2	87.7	87.7	85.8	87.0
0.9	87.2	87.1	86.9	87.0	87.6	87.7	87.2
0.95	87.6	87.6	87.1	84.1	86.9	86.8	86.7
Max	–	–	–	–	–	–	97.76
Avg	–	–	–	–	–	–	97.75

For pooling size 4, Adaptive Pooling remained inferior to traditional methods on both datasets. On CIFAR-10, the highest accuracy reached only 30.1% at alpha = 0.15, which was substantially lower than max pooling (60.1%) and average pooling (53.9%). On MNIST, the best performance was achieved at alpha = 0.35 with an accuracy of 87.0%, but this still trailed max pooling (97.76%) and average pooling (97.75%) by approximately 10%.

These results indicate that Adaptive Pooling remains highly sensitive to alpha selection and dataset characteristics and has not yet matched conventional pooling. Nevertheless, the findings provide valuable insights and motivate further development through architectural enhancements or integration with attention mechanisms.

To provide a clearer overview of the results, Table 8 presents a comprehensive summary of the optimal performance achieved by the proposed Adaptive Pooling method across different pooling sizes (2–4). This summary facilitates the observation of performance patterns and variations between datasets.

Table 8: Comparative accuracy of the proposed Adaptive Pooling method and existing pooling techniques (T-Max-Avg, Avg-TopK, Max, and Average Pooling) across different pooling sizes on the MNIST and CIFAR-10 datasets.

Dataset	Pool Size	The Best of α	Advt Pooling (%)	T-Max-Avg (%)	Avg-TopK (%)	Max (%)	Avg (%)
MNIST	2	0.9	64.9	98.7 (T = 0.8)	98.6	98.6	98.4
MNIST	3	0.1	79.8	98.9 (T = 0.8)	98.7	98.6	98.6
MNIST	4	0.5	87.2	98.3 (T = 0.8)	98.3	97.8	97.8
Cifar-10	2	0.3	25.2	61.5 (T = 0.7)	59	58.1	50
Cifar-10	3	0.1	26.5	63.8 (T = 0.7)	62.6	60.4	55
Cifar-10	4	0.15	30.1	64.4 (T = 0.7)	64.1	60.1	53.9

Table 8 compares the proposed Adaptive Pooling method with T-Max-Avg, Avg-TopK, max pooling, and average pooling on MNIST and CIFAR-10 using pooling sizes from 2 × 2 to 4 × 4.

On MNIST, Adaptive Pooling showed improved performance with larger pooling sizes, with accuracy increasing from 64.9% to 87.2% and the performance gap with the best method decreasing from 33.8% to 11.1%. In contrast, on CIFAR-10, accuracy remained low across all configurations (25.2%–30.1%), approximately 34%–37% below T-Max-Avg.

These results indicate that Adaptive Pooling is more suitable for simple datasets such as MNIST but remains less effective for complex data like CIFAR-10. Nevertheless, the findings underscore the importance of the alpha parameter and motivate further development toward nonlinear mechanisms or a trainable alpha to improve generalization.

5 Additional Evaluation and Visualization Analysis of Adaptive Pooling Model Performance on Different CNN Architectures

This section presents an additional evaluation to assess the stability and consistency of the proposed Adaptive Pooling method when applied to a convolutional neural network (CNN) architecture different from LeNet-5. The main objective of this experiment is to examine whether the performance of the proposed method remains stable in models with greater depth and higher parameter complexity.

The new CNN architecture used in this evaluation consists of three hierarchical convolutional blocks, each followed by Batch Normalization and a ReLU activation function to accelerate convergence and mitigate

the risk of vanishing gradients. Each block is concluded with a 2 × 2 MaxPooling layer to reduce spatial resolution while retaining essential features.

After the three convolutional blocks, an Adaptive Pooling layer with an output size of 4 × 4 is added as the main focus of this study. This layer employs the α parameter (ranging from 0 to 1) to balance the contribution between average pooling and max pooling adaptively. In the final stage, the network includes two fully connected layers with 256 and 128 neurons, respectively, each followed by a Dropout rate of 0.5 to reduce overfitting, and a Softmax layer for multi-class classification.

A schematic illustration of the architecture and its detailed configuration are shown in Fig. 6 and Table 9, respectively.

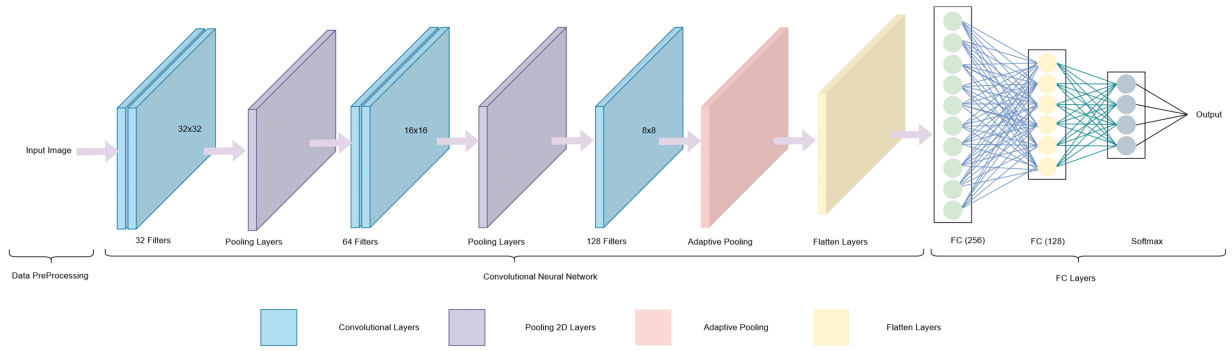


Figure 6: CNN model architecture used for testing the proposed Adaptive Pooling method.

Table 9: Network architecture.

Layer	Layer Type	Size of Feature Map	Kernel Size	Param	Activation Function
Input Layer	Input	32 × 32 × 3	–	0	–
Conv1	Convolution	32 × 32 × 32	3 × 3	896	ReLU
Conv2	Convolution	32 × 32 × 32	3 × 3	9248	ReLU
MaxPool1	Max Pooling	64 × 64 × 32	2 × 2	0	–
Conv3	Convolution	16 × 16 × 64	3 × 3	18,496	ReLU
Conv4	Convolution	16 × 16 × 64	3 × 3	36,928	ReLU
MaxPool2	Max Pooling	8 × 8 × 64	2 × 2	0	–
Conv5	Convolution	8 × 8 × 128	3 × 3	73,856	ReLU
AdtvPool	Adaptive Pooling	4 × 4 × 128	–	1	- (α)
Flatten	Flatten	2048	–	0	
FC1	Fully Connected	256	–	524,544	ReLU
FC2	Fully Connected	128	–	32,896	ReLU
FC3	Fully Connected	10	–	1290	Softmax

The experiments were conducted using the same training configuration as in the previous section, with 50 epochs, a batch size of 16, and the Adam optimizer initialized with a learning rate of 0.000001. The main difference lies in the pooling size, which was set to 4 × 4, based on the earlier results in this study, indicating that Adaptive Pooling achieved its highest accuracy at this configuration. Therefore, all evaluations conducted on the new CNN architecture employed a 4 × 4 pooling size to maintain consistency and ensure a fair comparison across architectures.

Table 10 presents the accuracy results obtained from training the Adaptive Pooling model on the MNIST dataset using various α parameter values (ranging from 0.1 to 0.95).

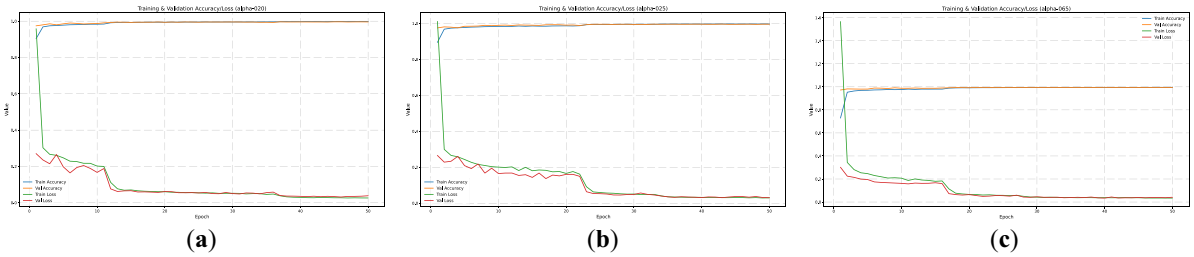
Table 10: Experimental results of Adaptive Pooling with a pooling size of 4 × 4 on the MNIST dataset using a different CNN architecture.

α Value	Exp-1 (%)	Exp-2 (%)	Exp-3 (%)	Exp-4 (%)	Exp-5 (%)	Exp-6 (%)	Average (%)
0.1	99.72	99.67	99.59	99.66	99.63	99.61	99.65
0.15	99.59	99.67	99.63	99.60	99.60	99.65	99.62
0.2	99.67	99.61	99.67	99.71	99.66	99.63	99.66
0.25	99.68	99.66	99.64	99.68	99.65	99.63	99.66
0.3	99.65	99.68	99.60	99.64	99.60	99.61	99.63
0.35	99.72	99.70	99.66	99.58	99.59	99.62	99.65
0.4	99.46	99.66	99.67	99.62	99.67	99.62	99.62
0.45	99.70	99.63	99.65	99.66	99.62	99.61	99.65
0.5	99.61	99.58	99.68	99.67	99.66	99.60	99.63
0.55	99.68	99.64	99.66	99.65	99.73	99.57	99.66
0.6	99.69	99.60	99.68	99.68	99.66	99.70	99.67
0.65	99.67	99.67	99.63	99.63	99.67	99.63	99.65
0.7	99.61	99.67	99.58	99.63	99.58	99.55	99.60
0.75	99.66	99.61	99.65	99.63	99.68	99.64	99.65
0.8	99.72	99.65	99.71	99.53	99.68	99.62	99.65
0.85	99.66	99.69	99.63	99.61	99.69	99.65	99.66
0.9	99.74	99.61	99.63	99.62	99.62	99.56	99.63
0.95	99.58	99.69	99.68	99.70	99.62	99.64	99.65
Max	99.63	99.70	99.58	99.68	99.64	99.66	99.65
Avg	99.66	99.64	99.65	99.70	99.71	99.71	99.68

Table 10 reports the performance of the Adaptive Pooling method on the MNIST dataset using a pooling size of 4 × 4 across multiple α values ranging from 0.1 to 0.95. The results show that the model achieves highly consistent performance, with average accuracies between 99.60% and 99.68% across six independent runs. The performance differences among the α values are minimal, indicating that the model remains stable and is not strongly sensitive to α variation when applied to MNIST.

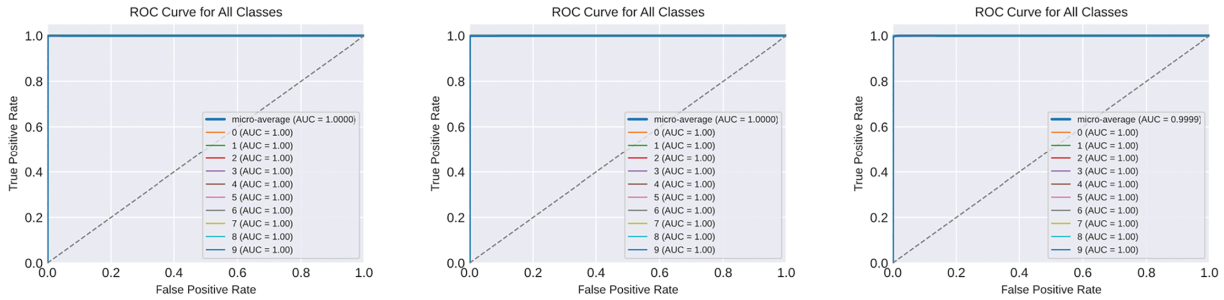
The highest average accuracy (99.7%) is obtained at α = 0.65, although several other α configurations yield comparable results. These findings suggest that the notable performance improvement observed in this experiment relative to the LeNet-based results in earlier tables is primarily influenced by the use of a deeper CNN architecture and its richer feature representation, rather than by the α parameter alone. Therefore, the results in Table 10 should be interpreted as an interaction effect between Adaptive Pooling and the underlying network architecture, rather than as evidence that Adaptive Pooling universally outperforms conventional pooling methods.

The performance visualization of the model is presented in Figs. 7–9 through the training curves, ROC curves, and confusion matrices. The ROC results show that all classes achieve an AUC value of 1.00, indicating perfect class separability and a highly stable decision boundary. This result is consistent with the confusion matrices, which exhibit almost no class-wise misclassification, demonstrating that the Adaptive Pooling model achieves near-perfect recognition performance on the MNIST dataset.



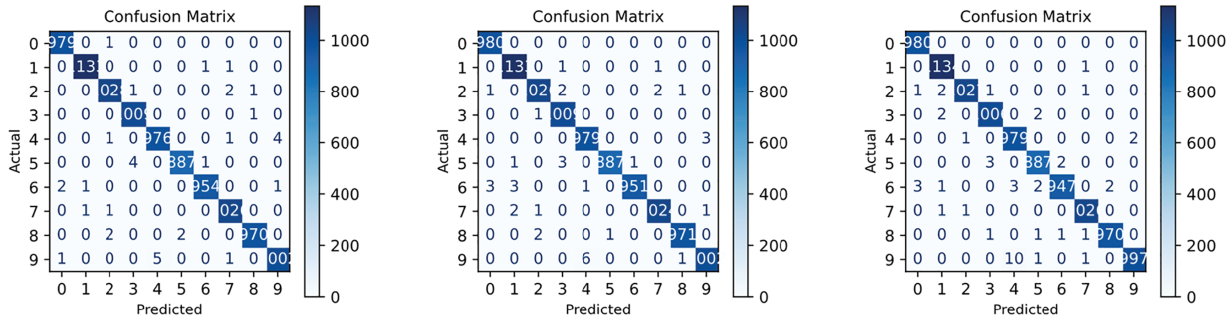
(**a**) (**b**) (**c**)

Figure 7: Model training curves of accuracy and loss validation in the MNIST Dataset. (**a**) a = 0.2, (**b**) b = 0.25, (**c**) c = 0.65.



(**a**) (**b**) (**c**)

Figure 8: Visualizing the ROC curve for the model in the MNIST Dataset. (**a**) a = 0.2, (**b**) a = 0.25, (**c**) a = 0.65.



(**a**) (**b**) (**c**)

Figure 9: Visualizing the confusion matrix for the model in the MNIST Dataset. (**a**) a = 0.2, (**b**) a = 0.25, (**c**) a = 0.65.

To complement the visual interpretation shown in Figs. 8–12, offer a rigorous quantitative foundation for understanding the ROC curve behavior, AUC values, and the class-wise prediction patterns reflected in the confusion matrix.

The True Positive Rate (TPR), also referred to as sensitivity, is defined in Eq. (4) as the proportion of correctly identified positive samples:

$$TPR = \frac{TP}{FP + FN} \tag{4}$$

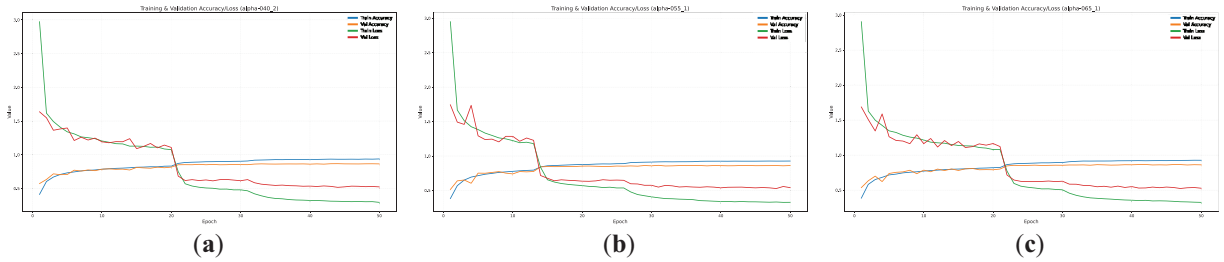


Figure 10: Model training curves of accuracy and loss validation in the CIFAR-10 Dataset. (**a**) a = 0.4, (**b**) a = 0.55, (**c**) a = 0.65.

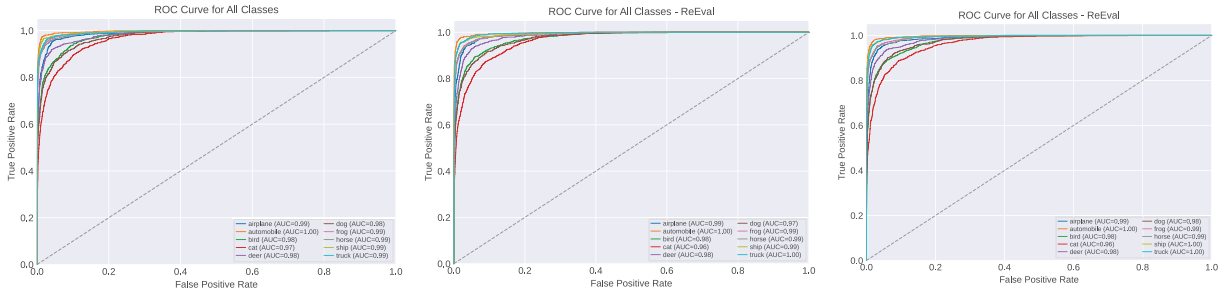


Figure 11: Visualizing the ROC curve for the model in CIFAR-10 Dataset. (**a**) a = 0.4, (**b**) a = 0.55, (**c**) a = 0.65.

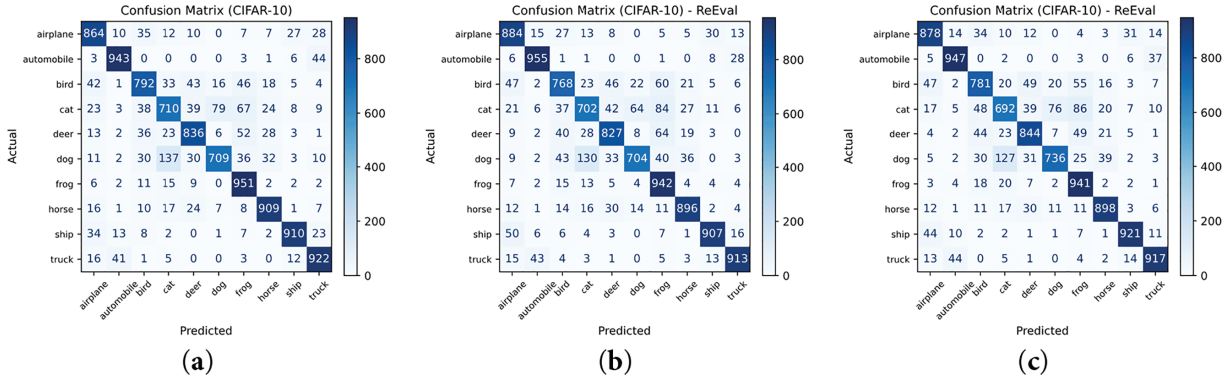


Figure 12: Visualizing the confusion matrix for the model in the CIFAR-10 Dataset. (**a**) a = 0.4, (**b**) a = 0.55, (**c**) a = 0.65.

The False Positive Rate (FPR), presented in Eq. (5), measures the proportion of negative samples incorrectly classified as positive:

$$FPR = \frac{FP}{FP + TN} \tag{5}$$

Using these definitions, the Receiver Operating Characteristic (ROC) curve can be expressed as shown in Eq. (6):

$$ROC = \{(FPR(t), TPR(t)) \mid t \in [0,1]\} \tag{6}$$

The Area Under the Curve (AUC), which summarizes the classifier's discriminative capability across all threshold values, is computed using Eq. (7):

$$AUC = \int_0^1 TPR\,(FPR)\,d\,(FPR) \tag{7}$$

For the multi-class classification scenario, the confusion matrix is defined in Eq. (8) as follows:

$$C_{i,j} = |\,\{x|y = i, \hat{y} = j\}\,| \tag{8}$$

Based on the confusion matrix, the overall classification accuracy can be obtained using Eq. (9):

$$Accuracy = \frac{\sum i\ \ C_{i,i}}{\sum i, j\ \ C_{i,j}} \tag{9}$$

These mathematical definitions support the visual evaluation results by providing formal, threshold-independent measures of the classifier's discriminative strength and class-wise consistency.

From the confusion matrix, it can be observed that the model achieves almost perfect predictions across all digit classes (0–9), with only a very small number of misclassified samples. The strong diagonal dominance indicates that the Adaptive Pooling mechanism supports stable and consistent feature extraction across classes.

These findings suggest that, when applied to a deeper CNN architecture, Adaptive Pooling with α = 0.65 is able to improve training stability and maintain high classification performance, achieving an accuracy of 99.7% and an AUC close to 1.00. Rather than indicating absolute superiority, these results demonstrate that the effectiveness of Adaptive Pooling is influenced by architectural depth and feature-representation capacity, which explains its stronger performance in this setting compared to LeNet-5.

In summary, this experiment confirms that employing Adaptive Pooling with a 4 × 4 pooling sizes not only yields high accuracy but also exhibits robustness against variations in the α parameter, making it a reliable approach for feature extraction in small-scale grayscale images such as MNIST.

Table 11 presents the accuracy results obtained from training the Adaptive Pooling model on the CIFAR-10 dataset with various α parameter values ranging from 0.1 to 0.95.

Table 11: Experimental results of Adaptive Pooling with a pooling size of 4 × 4 on the CIFAR-10 dataset using a different CNN architecture.

α Value	Exp-1 (%)	Exp-2 (%)	Exp-3 (%)	Exp-4 (%)	Exp-5 (%)	Exp-6 (%)	Average (%)
0.1	84.93	85.73	85.43	85.27	84.96	85.04	85.23
0.15	84.28	84.45	84.05	84.90	85.79	85.23	84.78
0.2	84.97	84.44	85.04	83.88	84.79	84.79	84.65
0.25	85.08	84.90	85.52	85.12	85.06	85.00	85.11
0.3	85.37	85.26	84.95	85.08	84.75	84.65	85.01
0.35	84.98	85.69	85.06	84.70	85.02	84.90	85.06
0.4	84.87	84.75	85.65	85.32	86.11	85.20	85.32
0.45	86.01	85.63	85.57	85.45	86.25	85.49	85.73
0.5	85.59	85.85	85.72	85.91	86.50	86.15	85.95
0.55	86.04	85.93	85.88	85.59	85.86	85.79	85.85

(Continued)

Table 11 (continued)

α Value	Exp-1 (%)	Exp-2 (%)	Exp-3 (%)	Exp-4 (%)	Exp-5 (%)	Exp-6 (%)	Average (%)
0.6	86.13	85.62	86.33	85.66	85.87	85.25	85.81
0.65	86.20	86.54	85.79	86.04	85.90	85.94	86.07
0.7	85.94	86.17	85.21	86.10	85.97	85.34	85.79
0.75	85.34	85.96	85.72	86.00	86.07	85.39	85.75
0.8	85.39	85.84	85.54	85.66	86.06	86.23	85.79
0.85	85.84	86.62	86.37	86.33	86.01	85.39	86.09
0.9	86.06	85.78	86.30	85.87	86.10	85.83	85.99
0.95	86.40	85.53	85.48	85.84	85.93	86.22	85.90
Max	84.46	83.58	84.60	84.89	84.40	85.33	84.54
Avg	86.63	86.23	86.01	85.92	85.79	86.45	86.17

Table 11 reports the experimental results of the Adaptive Pooling method on the CIFAR-10 dataset using a 4 × 4 pooling sizes across different α values. The average accuracy ranges from 85.0% to 86.2%, with relatively small fluctuations between repeated runs. This indicates that the Adaptive Pooling mechanism remains stable on a dataset with higher visual complexity compared to MNIST.

Although the absolute accuracy values are lower than those obtained on MNIST, the results show that the model maintains consistent behavior across a wide range of α values. This suggests that the performance of Adaptive Pooling is not driven by a single α configuration, but rather by its ability to preserve a balanced contribution between max pooling and average pooling in deeper CNN architectures.

The training curves in Fig. 10 illustrate the learning behavior of the Adaptive Pooling model on the CIFAR-10 dataset for three representative α values (0.40, 0.55, and 0.65). Across all configurations, the training and validation loss curves decrease steadily while the accuracy curves increase consistently, indicating a stable optimization process without oscillation or divergence. Although convergence in the validation accuracy occurs more gradually than in the MNIST experiments, no indications of overfitting or unstable gradient behavior are observed, even in deeper network settings. This suggests that the Adaptive Pooling mechanism supports smoother feature aggregation and contributes to more consistent convergence across different α values.

The Receiver Operating Characteristic (ROC) visualization in Fig. 11 further confirms the reliability of the model, where all classes achieve AUC values ≥ 0.96, reflecting a strong discriminative capability across categories such as airplane, automobile, bird, and cat. Meanwhile, the confusion matrices in Fig. 12 exhibit a predominantly diagonal pattern, indicating consistent class-wise prediction performance. The remaining misclassification cases mainly occur between classes with similar color and texture characteristics, such as bird vs. airplane and cat vs. dog, which are inherently difficult to separate at a resolution of 32 × 32 pixels.

These results further reinforce the conclusion that the Adaptive Pooling mechanism effectively balances local and global information through the α parameter, enabling the model to maintain high accuracy without compromising training stability, even when applied to deeper network architectures.

6 Results and Discussion

The additional experiments conducted on deeper CNN architectures show that the Adaptive Pooling method maintains stable and consistent performance across different α configurations. On the MNIST dataset, the model achieved accuracies between 0.996 and 0.997 across nearly all α values, indicating that

the method remains robust even when the contribution balance between max pooling and average pooling is varied. On the CIFAR-10 dataset, the accuracy ranged from 0.85 to 0.86 for α values between 0.45 and 0.65, which is consistent with earlier findings despite the higher visual complexity of this ten-class color dataset.

These results suggest that the α parameter functions as an effective balancing mechanism between dominant feature selection and contextual feature aggregation. Around α = 0.5, the model obtained an appropriate trade-off between sensitivity to salient features and robustness to noise, which led to better generalization on the test data. This trend is also reflected in the ROC curves and confusion matrices, which exhibit strong diagonal patterns and near-perfect AUC scores across most classes.

Furthermore, the 4 × 4 pooling configuration consistently produced better results than other pooling sizes, as it allows the network to preserve mid-level spatial structure while retaining sufficient global context, and at the same time reduces dimensionality before the fully connected layers.

Overall, these findings indicate that Adaptive Pooling with an adaptive α parameter achieves high accuracy while preserving architectural flexibility and without increasing model complexity. This shows that the proposed pooling mechanism is suitable for integration into different CNN architectures and datasets with diverse spatial characteristics. To further validate its cross-architecture behavior, a comparative analysis was also performed on LeNet-5, a Custom Deep CNN, and ResNet using both MNIST and CIFAR-10, as summarized in Table 12.

Table 12: Comparative performance of adaptive pooling across CNN architectures.

Dataset	Architecture	Pooling Type	Avg Best Accuracy (%)
Cifar-10	LeNet5	Avg-TopK Pooling	64.1 (K = 6)
		T-Max-Avg Pooling	64.4 (K = 6, T = 0.7)
		Max Pooling	60.1
		Avg Pooling	53.9
		Advt Pooling (α = 0.15, Ps = 4)	30.1
	Custom CNN	Avg-TopK Pooling	78.9 (K = 6)
		T-Max-Avg Pooling	79.1 (K = 6, T = 0.7)
		Max Pooling	84.5
		Avg Pooling	86.2
		Advt Pooling (α = 0.65, Ps = 4)	86.1
	ResNet-18	Avg-TopK Pooling	37.1 (K = 6)
		T-Max-Avg Pooling	40.2 (K = 6, T = 0.7)
		Global Avg Pooling	89.0
		Advt Pooling (α = 0.65, Ps = 4)	37.2
MNIST	LeNet5	Avg-TopK Pooling	98.3 (K = 3)
		T-Max-Avg Pooling	98.3 (K = 10, T = 0.8)
		Max Pooling	97.8
		Avg Pooling	97.8
		Advt Pooling (α = 0.5, Ps = 4)	87.2
	Custom CNN	Avg-TopK Pooling	98.9 (K = 3)
		T-Max-Avg Pooling	99.5 (K = 10)
		Max Pooling	99.6

(Continued)

Table 12 (continued)

Dataset	Architecture	Pooling Type	Avg Best Accuracy (%)
		Avg Pooling	99.7
		Advt Pooling (α = 0.65, Ps = 4)	99.7
	ResNet-18	Avg-TopK Pooling	99.2 (K = 3)
		T-Max-Avg Pooling	99.4 (K = 10)
		Global Avg Pooling	99.1
		Advt Pooling (α = 0.65, Ps = 4)	99.1

As shown in Table 12, a clear performance gap is observed between shallow and deeper architectures. On LeNet-5, Adaptive Pooling yields lower accuracy than Max/Average Pooling (and also lower than Avg-TopK and T-Max-Avg), especially on CIFAR-10, suggesting that limited feature abstraction in shallow networks can cause the mixed aggregation to suppress strong discriminative activations preserved by Max Pooling.

In contrast, on the Custom CNN, Adaptive Pooling becomes comparable to the baseline pooling methods and remains competitive with Avg-TopK and T-Max-Avg, indicating that deeper hierarchical representations better exploit the α-controlled balance between salient feature selection and contextual aggregation. However, on ResNet-18, replacing the default Global Average Pooling with spatial pooling (including Adaptive Pooling) degrades CIFAR-10 performance, implying that the effect depends not only on architecture depth but also on the head/downsampling design and the pooling insertion point.

These findings clarify the discrepancy observed in the earlier experiments: the gains reported in Tables 10 and 11 are driven by the presence of mid-level and high-level feature abstractions that are absent in shallow models, rather than by architectural replacement alone. Overall, Adaptive Pooling should therefore be regarded not as a universal substitute for Max or Average Pooling, but as a complementary pooling mechanism that becomes increasingly effective in deeper CNNs where feature hierarchies are more expressive.

Taken together, the cross-architecture analysis confirms that the α parameter acts as an adaptive controller between local dominant feature selection and global contextual aggregation, enabling training stability, competitive performance, and consistent generalization across datasets and architectures.

Acknowledgement: The author would like to sincerely thank the Beijing University of Chemical Technology for awarding the Presidential Scholarship, which has supported the author's tuition and academic expenses throughout this study. Special thanks also go to Professor Wang Kunfeng for his invaluable guidance and support.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: The authors confirm their contribution to the article as follows: Conceptualization, Nahdi Saubari and Kunfeng Wang; methodology, Nahdi Saubari; software, Nahdi Saubari; validation, Nahdi Saubari, Kunfeng Wang and Rachmat Muwardi; formal analysis, Nahdi Saubari; investigation, Nahdi Saubari; resources, Kunfeng Wang and Rachmat Muwardi; data curation, Nahdi Saubari; writing—original draft preparation, Nahdi Saubari; writing, review and editing, Kunfeng Wang, Rachmat Muwardi and Andri Pranolo; visualization, Nahdi Saubari; supervision, Kunfeng Wang; project administration, Kunfeng Wang; funding acquisition, Nahdi Saubari, Kunfeng Wang and Rachmat Muwardi. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: Available by request.

Ethics Approval: Not applicable.

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

References

1. Jie HJ, Wanda P. RunPool: a dynamic pooling layer for convolution neural network. Int J Comput Intell Syst. 2020;13(1):66–76. doi:10.2991/ijcis.d.200120.002.
2. Younesi A, Ansari M, Fazli M, Ejlali A, Shafique M, Henkel J. A comprehensive survey of convolutions in deep learning: applications, challenges, and future trends. IEEE Access. 2024;12:41180–218. doi:10.1109/access.2024.3376441.
3. Ersavas T, Smith MA, Mattick JS. Novel applications of convolutional neural networks in the age of transformers. Sci Rep. 2024;14(1):10000. doi:10.1038/s41598-024-60709-z.
4. Aloysius N, Geetha M. A review on deep convolutional neural networks. In: Proceedings of the 2017 International Conference on Communication and Signal Processing (ICCSP); 2017 Apr 6–8; Chennai, India. p. 588–92. doi:10.1109/ICCSP.2017.8286426.
5. Zafar A, Aamir M, Mohd Nawi N, Arshad A, Riaz S, Alruban A, et al. A comparison of pooling methods for convolutional neural networks. Appl Sci. 2022;12(17):8643. doi:10.3390/app12178643.
6. Sabri N. A comparison between average and max-pooling in convolutional neural network for scoliosis classification. Int J Adv Trends Comput Sci Eng. 2020;1(1.4):689–96. doi:10.30534/ijatcse/2020/9791.42020.
7. Zhao X, Wang L, Zhang Y, Han X, Deveci M, Parmar M. A review of convolutional neural networks in computer vision. Artif Intell Rev. 2024;57(4):99. doi:10.1007/s10462-024-10721-6.
8. Galanis NI, Vafiadis P, Mirzaev KG, Papakostas GA. Convolutional neural networks: a roundup and benchmark of their pooling layer variants. Algorithms. 2022;15(11):391. doi:10.3390/a15110391.
9. Nirthika R, Manivannan S, Ramanan A, Wang R. Pooling in convolutional neural networks for medical image analysis: a survey and an empirical study. Neural Comput Appl. 2022;34(7):5321–47. doi:10.1007/s00521-022-06953-8.
10. Song Z, Liu Y, Song R, Chen Z, Yang J, Zhang C, et al. A sparsity-based stochastic pooling mechanism for deep convolutional neural networks. Neural Netw. 2018;105(1):340–5. doi:10.1016/j.neunet.2018.05.015.
11. Zhao L, Zhang Z. A improved pooling method for convolutional neural networks. Sci Rep. 2024;14(1):1589. doi:10.1038/s41598-024-51258-6.
12. Özdemir C. Avg-topk: a new pooling method for convolutional neural networks. Expert Syst Appl. 2023;223:119892. doi:10.1016/j.eswa.2023.119892.
13. Zafar A, Saba N, Arshad A, Alabrah A, Riaz S, Suleman M, et al. Convolutional neural networks: a comprehensive evaluation and benchmarking of pooling layer variants. Symmetry. 2024;16(11):1516. doi:10.3390/sym16111516.
14. Shadoul I, Al-Hmouz R, Hossen A, Mesbah M, Deveci M. The effect of pooling parameters on the performance of convolution neural network. Artif Intell Rev. 2025;58(9):271. doi:10.1007/s10462-025-11273-z.
15. Williams T, Li RY. Wavelet pooling for convolutional neural networks. In: Proceedings of the 6th International Conference on Learning Representations; 2018 Apr 30–May 3; Vancouver, BC, Canada. p. 1–12.
16. Stergiou A, Poppe R. AdaPool: exponential adaptive pooling for information-retaining downsampling. IEEE Trans Image Process. 2023;32:251–66. doi:10.1109/tip.2022.3227503.
17. Stergiou A, Poppe R, Kalliatakis G. Refining activation downsampling with SoftPool. In: Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV); 2021 Oct 10–17; Montreal, QC, Canada. p. 10337–46. doi:10.1109/iccv48922.2021.01019.
18. Erak O, Alhussein O, Abou-Zeid H, Bennis M, Muhaidat S. Adaptive token merging for efficient transformer semantic communication at the edge. arXiv:2509.09955. 2025.
19. Jeanneret G, Simon L, Jurie F. Disentangling visual transformers: patch-level interpretability for image classification. In: Proceedings of the 2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW); 2025 Jun 11–12; Nashville, TN, USA. p. 2670–80. doi:10.1109/cvprw67362.2025.00252.

20. Tan M, Le Q. EfficientNetV2: smaller models and faster training. In: Proceedings of the 38th International Conference on Machine Learning; 2021 Jul 18–24; Online. p. 10096–106.
21. Yu W, Luo RM, Zhou P, Si C, Zhou Y, Wang X, et al. MetaFormer is actually what you need for vision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2022 Jun 21–24; New Orleans, LA, USA. p. 10819–29.
22. Hyun J, Seong H, Kim E. Universal pooling—a new pooling method for convolutional neural networks. Expert Syst Appl. 2021;180:115084. doi:10.1016/j.eswa.2021.115084.
23. Gao Z, Wang L, Wu G. LIP: local importance-based pooling. Int J Comput Vis. 2023;131(1):363–84. doi:10.1007/s11263-022-01707-4.
24. Wang X, Kang M, Chen Y, Jiang W, Wang M, Weise T, et al. Adaptive local cross-channel vector pooling attention module for semantic segmentation of remote sensing imagery. Remote Sens. 2023;15(8):1980. doi:10.3390/rs15081980.
25. Ferrà A, Aguilar E, Radeva P. Multiple wavelet pooling for CNNs. In: Computer vision—ECCV 2018 workshops. Cham, Switzerland: Springer; 2019. p. 671–5. doi:10.1007/978-3-030-11018-5_55.
26. Bergner B, Lippert C, Mahendran A. Token cropr: faster vits for quite a few tasks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2025 Jun 11–15; Nashville, TN, USA. p. 9740–50.
27. Lionnie R, Andika J, Alaydrus M. A new approach to recognize faces amidst challenges: fusion between the opposite frequencies of the multi-resolution features. Algorithms. 2024;17(11):529. doi:10.3390/a17110529.
28. Xu J, Li Z, Chen J. Feature extraction algorithm based on maximizing circular margin discriminant analysis. Concurr Comput. 2025;37(18–20):e70180. doi:10.1002/cpe.70180.
29. El-Khamy S, El-Bana S, Al-Kabbany A, Elragal H. Toward better semantic segmentation by retaining spectral information using matched wavelet pooling. Neural Comput Appl. 2025;37(10):7049–66. doi:10.1007/s00521-025-11008-9.
30. Kamada S, Ichimura T. An object detection by using adaptive structural learning of deep belief network. In: 2019 International Joint Conference on Neural Networks (IJCNN); 2019 Jul 14–19; Budapest, Hungary. p. 1–8. doi:10.1109/ijcnn.2019.8852145.
31. Rangel G, Cuevas-Tello JC, Nunez-Varela J, Puente C, Silva-Trujillo AG. A survey on convolutional neural networks and their performance limitations in image recognition tasks. J Sens. 2024;2024(1):2797320. doi:10.1155/2024/2797320.
32. Saleem MA, Senan N, Wahid F, Aamir M, Samad A, Khan M. Comparative analysis of recent architecture of convolutional neural network. Math Probl Eng. 2022;2022:7313612. doi:10.1155/2022/7313612.
33. Rodriguez-Martinez I, da Cruz Asmus T, Dimuro GP, Herrera F, Takáč Z, Bustince H. Generalizing max pooling via (a, b)-grouping functions for Convolutional Neural Networks. Inf Fusion. 2023;99:101893. doi:10.1016/j.inffus.2023.101893.
34. Akgül İ. A pooling method developed for use in convolutional neural networks. Comput Model Eng Sci. 2024;141(1):751–70. doi:10.32604/cmes.2024.052549.
35. Yang J, Chen F, Das RK, Zhu Z, Zhang S. Adaptive-avg-pooling based attention vision transformer for face anti-spoofing. In: Proceedings of the 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP); 2024 Apr 14–19; Seoul, Republic of Korea. p. 3875–9. doi:10.1109/ICASSP48485.2024.10446940.
36. LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. Proc IEEE. 1998;86(11):2278–324. doi:10.1109/5.726791.
37. Li H, Yue X, Meng L. Enhanced mechanisms of pooling and channel attention for deep learning feature maps. PeerJ Comput Sci. 2022;8(5):e1161. doi:10.7717/peerj-cs.1161.
38. Kumar T, Brennan R, Mileo A, Bendechache M. Image data augmentation approaches: a comprehensive survey and future directions. IEEE Access. 2024;12:187536–71. doi:10.1109/ACCESS.2024.3470122.
39. Cai G. Advanced image classification using convolutional neural networks. Sci Technol Eng Chem Environ Prot. 2024;2(9):1–15. doi:10.61173/117trk02.
40. Lukman A, Saputro WT, Seniwati E. Improving convolutional neural networks performance using modified pooling function. Matrik. 2024;23(2):343–52. doi:10.30812/matrik.v23i2.3763.