http://communications.science.ankara.edu.tr

Commun.Fac.Sci.Univ.Ank.Ser. A2-A3
Volume 66, Number 1, Pages 95–117 (2024)
DOI:10.33769/aupse.1356138
ISSN 1303–6009 E-ISSN 2618–6462

Research Article; Received: September 06, 2023; Accepted: November 17, 2023

WHICH POOLING METHOD IS BETTER: MAX, AVG, OR CONCAT (MAX, AVG)

Yahya DOGAN[1]
[1]Computer Engineering Department, Faculty of Engineering,
Siirt University, Siirt, TÜRKİYE

Abstract. Pooling is a non-linear operation that aggregates the results of a given region to a single value. This method effectively removes extraneous details in feature maps while keeping the overall information. As a result, the size of feature maps is reduced, which decreases computing costs and prevents overfitting by eliminating irrelevant data. In CNN models, the max pooling and average pooling methods are commonly utilized. The max pooling selects the highest value within the pooling area and aids in preserving essential features of the image. However, it ignores the other values inside the pooling region, resulting in a significant loss of information. The average pooling computes the average values within the pooling area, which reduces data loss. However, by failing to emphasize critical pixels in the image, it may result in the loss of significant features. To examine the performance of pooling methods, this study comprised the experimental analysis of multiple models, i.e. shallow and deep, datasets, i.e. Cifar10, Cifar100, and SVHN, and pool sizes, e.g. $2x2$, $3x3$, $10x10$. Furthermore, the study investigated the effectiveness of combining two approaches, namely Concat (Max, Avg), to minimize information loss. The findings of this work provide an important guideline for selecting pooling methods in the design of CNNs. The experimental results demonstrate that pooling methods have a considerable impact on model performance. Moreover, there are variances based on the model and pool size.

1. Introduction

Deep learning has achieved remarkable results in a variety of tasks [1–3]. There are various successful architectures in this field [4–6], and convolutional neural networks (CNNs) are widely utilized, particularly in image classification and object recognition. CNN architectures typically consist of convolutional, pooling, and fully connected layers. Convolutional layers perform calculations by sliding learnable filters over the data to extract diverse features. Each filter is adjusted to recognize a specific feature and is applied to the entire dataset, allowing feature maps to be

Keywords. Pooling, deep learning, convolutional neural network.
yahyadogan@siirt.edu.tr-Corresponding author; 0000-0003-1529-6118.

created. Pooling layers reduce the size of feature maps by performing subsampling in the relevant region, thereby diminishing the network's computational cost. Fully-connected layers are used to generate the network's output. The feature maps acquired from the convolution layers are vectorized and then passed through one or more fully-connected layers to yield the network output.

Pooling layers are crucial in CNN architecture. These layers reduce the dimensionality of a feature map by obtaining a summary of a given region, but it results in information loss. As a result, selecting the proper method for pooling is critical for model performance. In CNNs, max pooling and average pooling methods are extensively utilized, each having its advantages and disadvantages. The max pooling takes the maximum activation to represent the pooling region of interest. This approach eliminates other features by focusing on the most important elements, resulting in a more specific feature map. It is very sensitive to the direction, size, and position of items in a given feature map. The average pooling, on the other hand, uses the average of all features to represent the region of interest in the feature map, allowing for the creation of a more generic feature map.

It is unclear which pooling method performs best under different conditions. In this study, different model architectures, i.e. shallow and deeper, different pooling sizes, e.g. $2x2$, $3x3$, ..., $10x10$, and different datasets, i.e. Cifar10 [7], Cifar100 [7], SVHN [8], were compared to evaluate the performance of the methods. Furthermore, the effect of concatenating these approaches to capture both significant features in images and overall patterns in data was experimentally studied.

The rest of the article is organized as follows. Section 2 is a brief review of previous research on pooling layers. Section 3 discusses the materials and processes in detail. Section 4 describes experimental studies and results. The article concludes with future directions.

2. Related Works

In general, two common pooling methods are utilized for reducing the size of feature maps: local pooling and global pooling. In local pooling, to minimize the dimensionality of the feature map, inferences are drawn from small neighboring regions within the feature map, e.g., $3x3$. In contrast, global pooling generates a single scalar value that represents the entire feature map. This research focuses on local pooling methods. Numerous studies have been conducted in this area since local pooling methods have a substantial impact on the success of CNNs. The studies can be categorized into four primary categories: value-based, probability-based, rank-based, and transformed-based methods [9].

In value-based pooling methods [10–12, 14–16], a value selection is determined based on a criterion among the pooling region's values. Mixed pooling [10] adds a parameter to choose between maximum and average pooling. Detail preservation pooling [12] is an adaptive pooling method that uses an inverse bilateral filter to amplify local spatial changes while retaining key structural details. It includes a

learnable parameter for controlling the feature map's downsampling. Spatial pyramid pooling [13] creates fixed-length outputs regardless of input size and reduces information loss due to cropping. LEAP pooling [14] employs a shared linear filter for each feature channel to combine features in the pooling region, resulting in a reduction in the number of parameters and training errors. Dynamic correlation pooling [15] introduces a correlation pooling technique that relies on the Mahalanobis distance between adjacent pixels in an image. The output is dynamically determined by assessing the relationship between the Mahalanobis distance and a predefined threshold distance. The avg-topk pooling [16] method takes the average of the top-k activations in the pooling area, assisting in the preservation of significant features and addressing the problem caused by outliers and noises.

Probability-based pooling methods [17–21] calculate the probability of trading off between max and average pooling, thereby reducing error rates and preventing overfitting. Lp pooling [17] determines the pooling type based on a probability value P. $P = 1$ corresponds to Gaussian mean, while $P = \infty$ corresponds to maximum pooling. Stochastic pooling [18] substitutes a stochastic procedure for deterministic pooling operations. Within this approach, activations within the pooling region undergo normalization and are randomly selected through the utilization of a multinomial distribution. The max pooling dropout [19] combines max pooling and dropout techniques, and it has been experimentally shown to outperform maximum and scaled maximum probabilities. Song et al. (2018) [20] propose a sparsity-based stochastic pooling method that balances the advantages of max and average pooling by utilizing the sparsity level and control function of activations to obtain a feature representation. Hybrid pooling [21] combines both the max and average pooling methods by calculating maximum and average pooling values for the given pooling region. This combination is performed using a predefined probability.

Rank-based pooling methods [22–24] rank the activations within a specified pooling region and produce a pooled output based on weighted activation sums. During training, the weights are often learned via the back-propagation approach. This strategy overcomes the scale issues encountered by value-based pooling methods, allowing the model to capture critical activations and perform better. These methods are categorized into three groups based on weighting mechanisms: rank-based average pooling (RAP), rank-based weighted pooling (RWP), and rank-based stochastic pooling (RSP). The RAP approach considers the greatest activations in the pooling region, ignoring the rest, and then computes the average of these activations. RAP has a superior discriminative ability and provides a balanced approach between maximum and average pooling. The RWP strategy recognizes that each region is not equally significant. It takes the weighted average of each activation in the specified pooling region multiplied by a suitable coefficient. In RWP, reasonable weights are ascribed to activations based on their magnitudes, with the largest activation receiving the highest weight and the smallest activation receiving the least weight. The RSP strategy substitutes traditional pooling operations

with a stochastic procedure in which activations are chosen based on probabilities derived from a multinomial distribution. RSP, as opposed to value-based stochastic pooling, computes probability based on activation order rather than activation value. The principal advantage of this strategy is the high degree of randomness in activation selection.

Domain-based pooling methods [25–27] use different domains to reduce spectral variance in feature maps, such as time, space, frequency, and wavelet domains. These studies often concentrate on the frequency domain and aim to filter out higher frequencies by removing low-frequency components. This transformation is achieved using various transformations such as Discrete Fourier Transform, Fast Fourier Transform, Hartley Transform, and Discrete Cosine Transform. These pooling methods can be sensitive to noise and perform filtering, but the computational cost can be high.

When various pooling approaches are compared across categories, it is clear that popular CNN models, e.g., VGG16 [28], ResNet [29], and EfficientNet [30], prefer the usage of max and average pooling methods due to their computational efficiency and lack of additional parameters. Therefore, in this study, the effects of these methods on multiple datasets, CNN models, pool sizes, and the combination of these methods have been investigated through experimental studies.

3. Material and Methods

3.1. **Pooling Methods.** In popular CNN models, average and max pooling methods are frequently utilized. In the average pooling, feature maps are divided into discrete rectangular regions, and sampling is performed by calculating the average activation value of each region. Mathematically, the expression for average pooling is as follows [31]:

$$f_{average} = \frac{1}{N} \sum_{i=1}^{N} x_i \tag{1}$$

Where x_i represents each activation value in the pooling area, and N denotes the number of activation values in that area. In max pooling, the activation within a pooling region with the highest value is chosen. This method is extensively used in CNN architectures and is reported to perform better in sparser encoding and simpler linear classifiers. As a consequence, its prominence has increased over the past few years. Mathematically, the expression for max pooling is as follows [34]:

$$f_{max}(x) = max_i(x_i) \tag{2}$$

Figure 1 depicts the (b) maximum pooling, (c) average pooling, and (d) Concat(Avg, Max) pooling outputs for a $4x4$ feature map (a) with a pool size of $2x2$.

Examining the efficacy of concatenating the two methods, i.e. ConCat (Avg, Max), is one of the primary contributions of this study. Extensive experimental

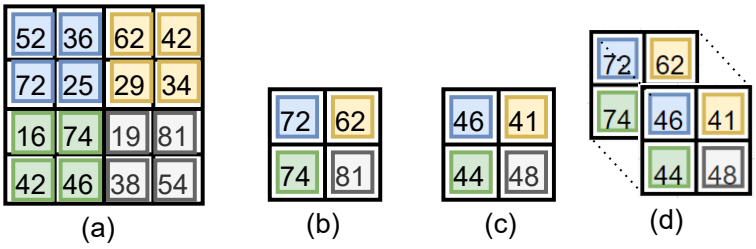

FIGURE 1. Illustration of (b) max/, (c) average pooling, and (d) Concat(Avg, Max) with a pooling area of size $2x2$ and stride of 2.

studies have been performed in this context to determine whether this method is required for small pooling sizes and whether it prevents information loss for large pooling sizes.

3.2. **Datasets.** Experimental studies were conducted using three benchmark datasets, namely CIFAR10, CIFAR100, and SVHN, to evaluate the performance of pooling methods. These datasets are commonly used to compare CNNs in the literature [9]. The CIFAR-10 dataset [7] is comprised of a total of $60,000$ RGB images with a resolution of $32x32$ and ten categories of labeling. The dataset consists of $60,000$ images divided into two sets: $50,000$ images for training and $10,000$ images for testing. Each class in the dataset has an equal number of examples, resulting in $6,000$ images per class. There exists a complete distinction between the classes. This particular dataset is frequently cited in the scientific literature for proposing new methodological approaches. The CIFAR100 dataset, which was introduced by Krizhevsky et al. (2014) [7], consists of 100 classes with 600 images in each class, totaling $60,000$ images. The images per class are separated into 500 training images and 100 test images. The resolution of the image is identical to CIFAR10, i.e., $32x32$. The dataset also has 20 superclasses in addition to the 100 classes. Consequently, each image has a "fine" label that corresponds to its class and a "coarse" label that corresponds to its superclass. The Street View House Numbers (SVHN) dataset [8] is a real-world image dataset commonly used to develop deep learning algorithms with minimal preprocessing and formatting requirements. It contains $600,000$ $32x32$ RGB images of printed numbers ranging from 0 to 9, serving as a number classification benchmark dataset. It is similar to MNIST, e.g., images composed of small cropped digits, but includes additional labeled data and tackles the more difficult, unsolved real-world problem of recognizing numbers and digits in images of natural scenes. The cropped images contain the digit of interest as well as adjacent digits and other distracting objects. Figure 2 depicted some examples taken from these datasets.

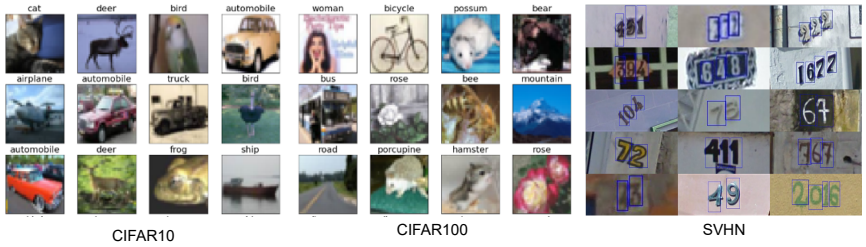

FIGURE 2. Random samples and classes from CIFAR10 (left), CIFAR100 (center), and SVHN (right).

3.3. **Models.** To assess the performance of pooling methods, two distinct models were utilized. Firstly, a shallow model based on the LeNet-5 [31] architecture was constructed to compare pooling methods. The architecture in question comprises a pair of convolutional layers, followed by two pooling layers, and finally, three fully connected (FC) layers. To accelerate convergence, the ReLU [32] activation function was used as opposed to the tanh activation function in the original LeNet-5. The first convolutional layer employed six learnable filters, while the second convolutional layer employed sixteen filters, both with a filter size of $5x5$. To preserve the resolution of the feature maps, a padding value of 2 was utilized in the convolutional layers. The stride value in the pooling layers was set equal to the kernel size to avoid overlapping. Figure 3 illustrates the LeNet-5-based architecture.

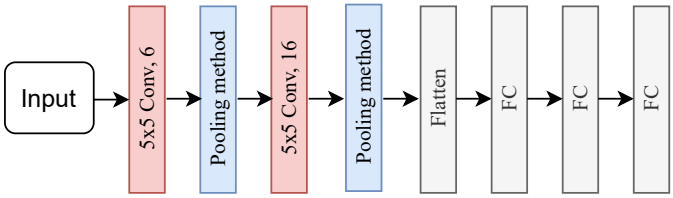

FIGURE 3. LeNet-5 based model.

For every method of pooling, the models were trained from scratch for 30 epochs. In other words, pre-trained weights were not used. As the loss function, cross-entropy was used, and stochastic gradient descent was employed as the optimization algorithm. [33]. The learning rate was set to $1e-3$, momentum was set to 0.9, and the batch size to 16.

Secondly, to compare pooling methods in a different and deeper model, a model based on ResNet-9 [29] was used. The model in question comprises eight convolutional layers, followed by four pooling layers, and finally, one FC layer. All convolutional layers utilized $3x3$ filters. After the convolutional layers, batch normalization layers were included to avoid the common vanishing gradient issue in

deep learning models. ReLU activation layers were applied after the respective layers. Two residual networks were utilized to facilitate the training of the deep model. A residual network is defined as a kind of neural network that includes skipping connections, which perform identity mappings and integrate layer outputs with the input via element-wise addition. Figure 4 depicts the specifics of the model developed based on ResNet-9. During the training phase, the LeNet-5 model's hyperparameters were utilized.

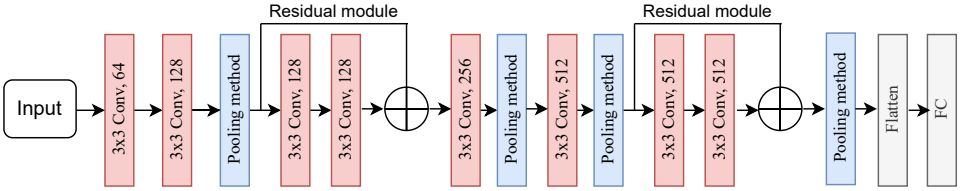

FIGURE 4. ResNet-9 based model.

3.4. **Metrics.** In this study, classification-based models were created and the performance of pooling methods was compared. A set of metrics known as the confusion matrix is utilized to assess the efficacy of classification models. The confusion matrix comprises four different concepts: True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN). A confusion matrix is typically represented in a table format as follows:

TABLE 1. Confusion matrix. TP: The number of in which the true class is positive and the model predicts it to be positive, FP: The number of in which the true class is negative but the model predicts it to be positive. TN: The number of instances in which the true class is negative and the model predicts it to be negative. FN: The number of instances in which the true class is positive but the model predicts it to be negative.

	Actually Positive	Actually Negative
Predicted Positive	TN	FP
Predicted Negative	FN	TP

To compare the performance of methods using the confusion matrix, 4 metrics were used: accuracy, precision, recall, and F1 score. The accuracy metric quantifies the proportion of instances correctly predicted by a model. This metric is used to evaluate a classification model's overall efficacy.

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{3}$$

The precision metric indicates how many of the values predicted as positive by a model are actually positive. This metric limits the number of false positive predictions made by a classification model.

$$Precision = \frac{TP}{(TP + FP)} \tag{4}$$

The recall metric indicates how many values that should be classified as positive are predicted as positive by a model. This metric is used to limit a classification model's number of false negative predictions.

$$Recall = \frac{TP}{(TP + FN)} \tag{5}$$

The F1 score metric is the harmonic mean of the metrics for precision and recall. This metric takes false positive and false negative predictions made by a classification model into account.

$$F1score = 2 * \frac{(Precision * Recall)}{(Precision + Recall)} \tag{6}$$

4. Experiment and Results

In this section, the performance of pooling methods was compared for different models (Lenet-5 and Resnet-9), different datasets (Cifar10, Cifar100, and SVHN), and different pool sizes ($2x2$, $3x3$, ..., $10x10$). In this context, firstly, the performance of methods for different pool sizes was examined using the Lenet-5 model and the Cifar10 dataset. For each pooling method, the Lenet-5 model was trained from scratch for 30 epochs. Figure 5 shows the training accuracy achieved for 4 different pool sizes, i.e., $2x2$, $5x5$, $7x7$, and $10x10$. Analyzing the graphs reveals that the average pooling begins with low accuracy for all pool sizes. In later epochs, it is observed that the average pooling achieves higher accuracy values for small pool sizes, e.g. $2x2$ and $5x5$, but falls behind other methods for larger pool sizes, such as $7x7$ and $10x10$.

Table 2 compares pooling methods quantitatively using the Lenet-5 model and the Cifar10 dataset. Examining Table 2 reveals that the average pooling method is more effective for small pool sizes, such as $2x2$, and $3x3$. Due to the fact that the stride value is equal to the kernel size, there is no overlap. In other words, as the size of the pool increases, there is an expected increase in information loss, resulting in a decrease in the accuracy of all models. As the pool size increases, it can be observed that the efficacy of the average pooling degrades more rapidly than other methods. Depending on the pool size, the max pooling outperforms the average pooling after a certain point, such as $6x6$. Within the scope of this study, the proposed Concat(Avg, Max) method demonstrated superiority over the other two methods as the pool size increased.

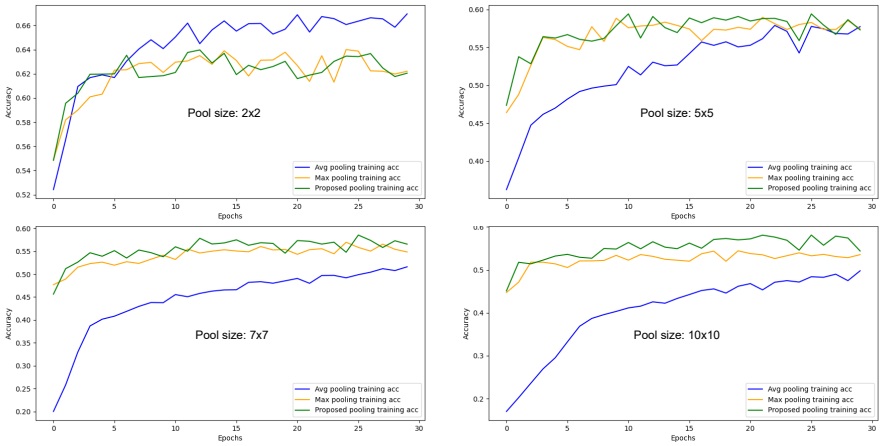

FIGURE 5. Training graphs of pooling methods for $2x2$, $5x5$, $7x7$, and $10x10$ pool sizes using the Lenet-5 model and the Cifar10 dataset.

In Figure 6, the performance of the aforementioned pooling methods for different pool sizes is presented graphically using the F1 score metric. As observed, the average pooling has a significant advantage over the other methods for a $2x2$ pool size. Interestingly, for a $5x5$ pool, all methods yield virtually identical results. It is evident that the Concat(Avg, Max) method is more successful for large pools ($\geq 6x6$). Since the average pooling takes the average of values in the pooling area, the influence of high activations diminishes as the pool size increases, resulting in a general performance decline.

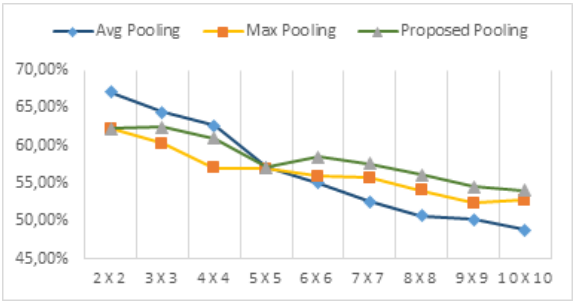

FIGURE 6. F1 score performances of pooling methods for varying pool sizes using the Lenet-5 model and the Cifar10 dataset.

Secondly, experimental studies were conducted to evaluate the performance of the methods on a more challenging dataset. In this context, the Lenet-5 model

TABLE 2. Compare pooling methods for Model: LeNet-5 Dataset: Cifar10

Pool size	Method	Accuracy	Precision	Recall	F1 score
2x2	Avg pooling	**0.6695**	**0.6763**	**0.6695**	**0.6707**
	Max pooling	0.6221	0.6304	0.6221	0.6230
	Concat (Avg-Max)	0.6205	0.6298	0.6205	0.6225
3x3	Avg pooling	**0.6454**	**0.6495**	**0.6454**	**0.6447**
	Max pooling	0.6087	0.6168	0.6087	0.6030
	Concat (Avg-Max)	0.6218	0.6366	0.6218	0.6250
4x4	Avg pooling	**0.6224**	**0.6382**	**0.6224**	**0.6267**
	Max pooling	0.5781	0.5874	0.5781	0.5705
	Concat (Avg-Max)	0.6151	0.6171	0.6151	0.6098
5x5	Avg pooling	**0.5773**	0.5749	**0.5773**	0.5710
	Max pooling	0.5743	0.5966	0.5743	0.5694
	Concat (Avg-Max)	0.5731	**0.5983**	0.5731	**0.5715**
6x6	Avg pooling	0.5600	0.5546	0.5600	0.5506
	Max pooling	0.5664	0.5857	0.5664	0.5594
	Concat (Avg-Max)	**0.5862**	**0.6065**	**0.5862**	**0.5848**
7x7	Avg pooling	0.5370	0.5360	0.5370	0.5254
	Max pooling	0.5614	0.5774	0.5614	0.5573
	Concat (Avg-Max)	**0.5786**	**0.5956**	**0.5786**	**0.5766**
8x8	Avg pooling	0.5160	0.5129	0.5160	0.5069
	Max pooling	0.5484	0.5670	0.5484	0.5409
	Concat (Avg-Max)	**0.5655**	**0.5814**	**0.5655**	**0.5615**
9x9	Avg pooling	0.5111	0.5063	0.5111	0.5023
	Max pooling	0.5360	0.5483	0.5360	0.5240
	Concat (Avg-Max)	**0.5495**	**0.5754**	**0.5495**	**0.5453**
10x10	Avg pooling	0.4979	0.4919	0.4979	0.4880
	Max pooling	0.5356	0.5446	0.5356	0.5281
	Concat (Avg-Max)	**0.5441**	**0.5732**	**0.5441**	**0.5409**

and the Cifar100 dataset were used to evaluate various pool sizes. Figure 7 shows the training accuracy achieved by the methods for $2x2$, $5x5$, $7x7$, and $10x10$ pool sizes. Similar to the Cifar10 dataset, it can be observed that the average pooling has lower initial accuracy than other methods for all pool sizes during the initial epochs. In later epochs, the average pooling performs better than other methods for small pool sizes, but it lags for large pool sizes. In contrast to the Cifar10 dataset, the methods in this dataset attain a high accuracy value for the $2x2$ pool size at the $5th$ epoch and then decline. This observation indicates that it is preferable to use the model from the epoch with the highest accuracy rather than the model obtained after the training process.

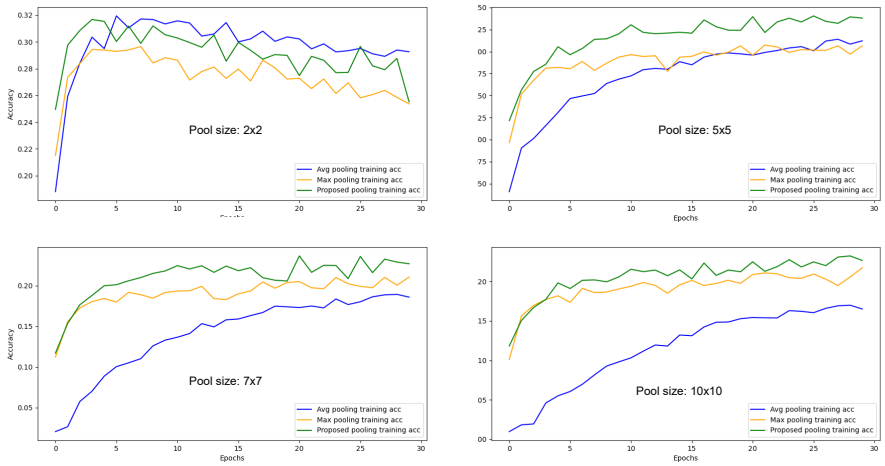

FIGURE 7. Training graphs of pooling methods for $2x2$, $5x5$, $7x7$, and $10x10$ pool sizes using the Lenet-5 model and the Cifar100 dataset.

Using the LeNet-5 model and the Cifar100 dataset, Table 3 illustrates quantitatively the efficacy of various pooling methods. Similar to the Cifar10 dataset, it is observed that the average pooling performs better for smaller pool sizes. Specifically, for a $2x2$ pool size, the closest performing method, i.e., max pooling, exhibits a 3.41% increase in performance. As the pool size increases, the efficacy of the average pooling diminishes relative to that of other methods. The Concat(Avg, Max) method has a 6.28% higher success rate than the average pooling when evaluating the $10x10$ pool size.

In Figure 8, the performance of different pooling methods for various pool sizes is graphically presented using the LeNet-5 model and the Cifar100 dataset. It can be observed that the average pooling yields the highest performance for a $3x3$ pool. In the Cifar10 dataset, the highest score was obtained for a $2x2$ pool size. This suggests that the performance of a model can vary based on the size of the data pool, even when using the same model and different datasets. Similar to the Cifar10 dataset, when the average pooling method is used and the pool size is increased, the model's success significantly decreases. Concat(Avg, Max) obtains the highest performance for larger pool sizes, while max pooling also achieves relatively close scores.

Cifar10 and Cifar100 datasets are partially similar datasets. In the continuation of the study, experimental work was conducted using the SVHN dataset, which contains more diverse images, and a Lenet-5-based model. Table 4 provides quantitative performance results for varied pool sizes. Similar to other datasets, the average pooling exhibited higher performance for smaller pool sizes. As the

TABLE 3. Compare pooling methods for Model: Lenet Dataset: Cifar100

Pool size	Method	Accuracy	Precision	Recall	F1 score
2x2	Avg pooling	**0.2926**	**0.2957**	**0.2926**	**0.2878**
	Max pooling	0.2536	0.2676	0.2536	0.2522
	Concat (Avg-Max)	0.2551	0.2676	0.2551	0.2537
3x3	Avg pooling	**0.3314**	0.3355	**0.3314**	**0.3228**
	Max pooling	0.2703	0.2804	0.2703	0.2588
	Concat (Avg-Max)	0.3026	**0.3382**	0.3026	0.2935
4x4	Avg pooling	**0.2949**	0.3052	0.2949	**0.2817**
	Max pooling	0.2594	0.2717	**0.2594**	0.2451
	Concat (Avg-Max)	0.2930	**0.3154**	0.2930	0.2762
5x5	Avg pooling	0.2264	0.2235	0.2264	0.2048
	Max pooling	0.2130	0.2188	0.2130	0.1903
	Concat (Avg-Max)	**0.2352**	**0.2670**	**0.2352**	**0.2167**
6x6	Avg pooling	0.2122	0.2040	0.2122	0.1901
	Max pooling	0.2065	0.2134	0.2065	0.1853
	Concat (Avg-Max)	**0.2380**	**0.2702**	**0.2380**	**0.2170**
7x7	Avg pooling	0.2042	0.1902	0.2042	0.1797
	Max pooling	0.2068	0.2071	0.2068	0.1845
	Concat (Avg-Max)	**0.2323**	**0.2597**	**0.2323**	**0.2105**
8x8	Avg pooling	0.1858	0.1841	0.1858	0.1636
	Max pooling	0.2105	0.2133	0.2105	0.1887
	Concat (Avg-Max)	**0.2268**	**0.2494**	**0.2268**	**0.2082**
9x9	Avg pooling	0.1836	0.1803	0.1836	0.1593
	Max pooling	0.2166	0.2185	0.2166	0.1944
	Concat (Avg-Max)	**0.2321**	**0.2428**	**0.2321**	**0.2089**
10x10	Avg pooling	0.1651	0.1654	0.1651	0.1424
	Max pooling	0.2171	0.2138	0.2171	0.1923
	Concat (Avg-Max)	**0.2265**	**0.2340**	**0.2265**	**0.2052**

pool size increased, the efficacy of the average method decreased more compared to other methods. The average method demonstrated roughly half the efficacy of the Concat(Avg, Max) method when evaluating the $10x10$ pool size. In general, for a shallow model, the average pooling performed better for smaller pool sizes, whereas max pooling, particularly Concat(Avg, Max), stood out as the pool size increased. Figure 9 illustrates the performance of methods for the Lenet-5 model and the SVHN dataset. The performance of the average pooling was comparable for $2x2$, $3x3$, and $4x4$ pool sizes. As with other datasets, its performance decreased substantially as the size of the pool grew. The Concat(Avg, Max) method obtained an F1 score of 72.61% for the $10x10$ pool size, while the average pooling achieved a performance of 29.41%. In general evaluation of the Lenet-5 model, or in other

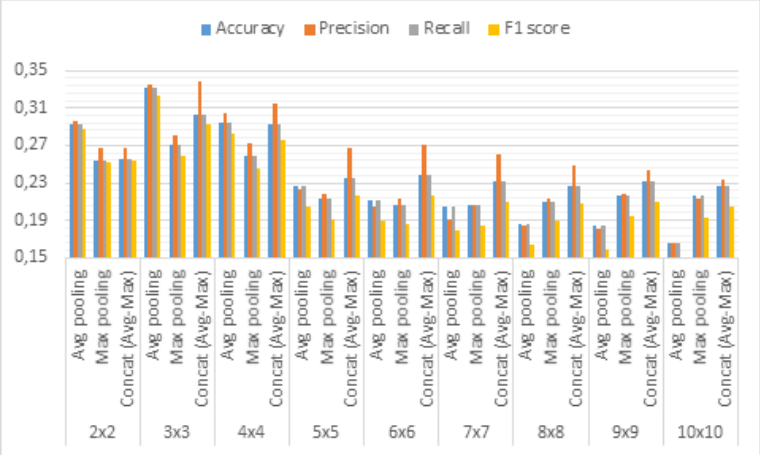

FIGURE 8. Accuracy, precision, recall, and F1 score results of pooling methods with various pool sizes in the LeNet model using the Cifar100 dataset.

words, a shallow model, the selection of pooling method is important depending on the pool size used. If the pool size is small, the average pooling should be preferred, whereas if the pool size is large, the maximum pooling method should be chosen. If the number of model parameters increase is not a concern, the Concat(Avg, Max) method can be applied to larger pool sizes.

In the continuation of the study, the performance of pooling methods was compared using ResNet-9 which has a deeper and different architecture. In this context, since the relevant model contained a greater number of pooling layers, the images were resized to a resolution of $224x224$. Figure 10 depicts the accuracy graphs acquired during the training phase for various pool sizes and each method. The shallow Lenet-5 model performed better with smaller pool sizes, such as $2x2$ and $3x3$, when the average pooling was applied to all the datasets used in the study. Figure 11 illustrates the efficacy of methods for various pool sizes utilizing the ResNet-9 model and the Cifar10 dataset. In addition, Table 5 quantifies all of the experimental results. In contrast to the Lenet-5 model, the average pooling in ResNet-9 trails behind other methods for all pool sizes. In addition, as the pool size increases, the performance of the average pooling method diminishes in both the shallow and deep models. While the max pooling and the Concat(Avg, Max) method attain comparable performance, the Concat(Avg, Max) method is observed to perform better for larger pool sizes.

Examined quantitatively, the max pooling for a $2x2$ pool attained a 13.12% higher F1 score than the average pooling. The difference between Concat(Avg, Max) and max pooling was less than 1%, and the max pooling was found to be superior. Considering a larger pool size of $5x5$, the Concat(Avg, Max) obtained the

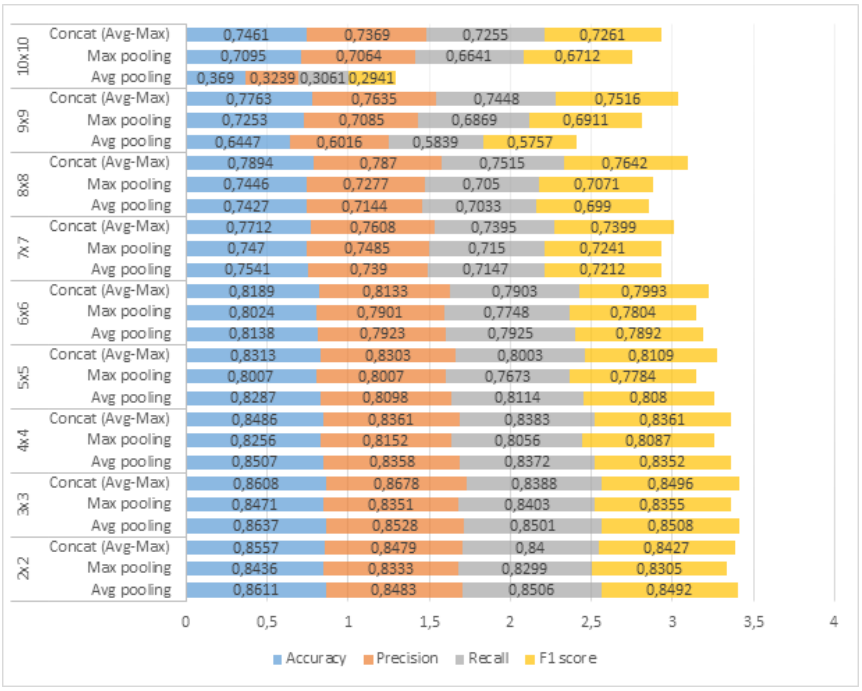

FIGURE 9. Accuracy, precision, recall, and F1 score results of pooling methods with various pool sizes in the LeNet-5 model using the SVHN dataset.

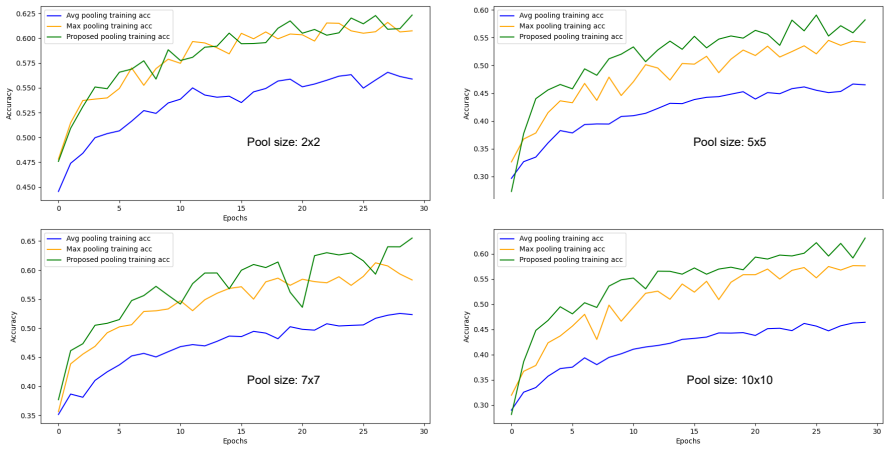

FIGURE 10. Training graphs of pooling methods for $2x2$, $5x5$, $7x7$, and $10x10$ pool sizes using the Resnet-9 model and the Cifar10 dataset.

best performance. It achieved a 21.3% improvement in the F1 score over the average pooling and a 7.44% improvement over the maximum pooling for this specific pool size. Notably, as the pool size increases, the methods' efficacy typically decreases due to information loss. Concat(Avg, Max) performed better than the $2x2$ pool size for $7x7$ and $8x8$ pool sizes. Due to the increased number of pooling layers in this experimental setting, the image resolution became inadequate; therefore, the stride value was fixed at 6 after a pool size of $6x6$.

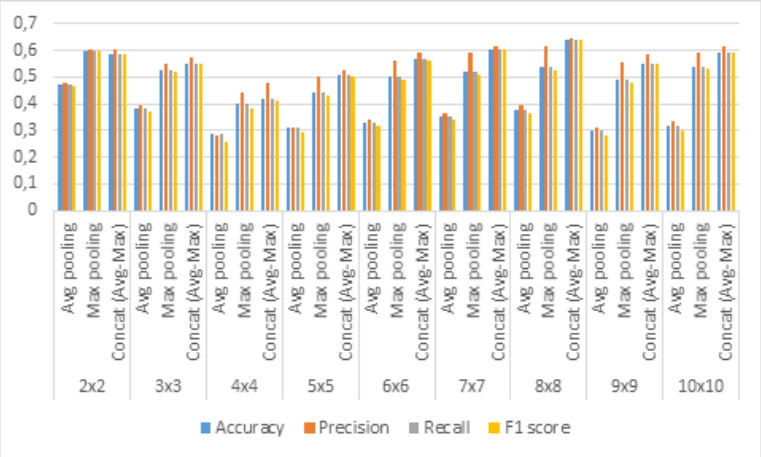

FIGURE 11. Accuracy, precision, recall, and F1 score results of pooling methods with various pool sizes in the ResNet-9 model using the Cifar10 dataset.

Using the ResNet-9 model and the Cifar100 dataset, experimental analysis was conducted to assess the performance of the methods on a similar but more difficult dataset. Table 6 provides a quantitative presentation of the results obtained for the respective experimental setting. In addition, to facilitate the comparison of methods for various pool sizes, the scores are presented in Figure 12 as a stacked bar graph. The graph reveals a situation comparable to the Cifar10 dataset. While the max pooling outperforms other methods for small pools, it is clear that the Concat(Avg, Max) method performs better as the pool size increases. Similar to the previous situation, $8x8$ and $9x9$ pool sizes received higher scores than $2x2$.

Finally, the performance of the methods was compared using the ResNet-9 model and the SVHN dataset. The quantitative results obtained for this experimental setup are provided in Table 7. Additionally, the F1 score of the methods for each pool size is given in Figure 13. In contrast to other experimental results, it was observed that even for the smallest pool size, the Concat(Avg, Max) method yielded higher scores. Notable is the fact that, for a $2x2$ pool size, the Concat(Avg, Max) method produced F1 score enhancements of 2.84% and 12.29%, compared to the maximum and average methods, respectively.

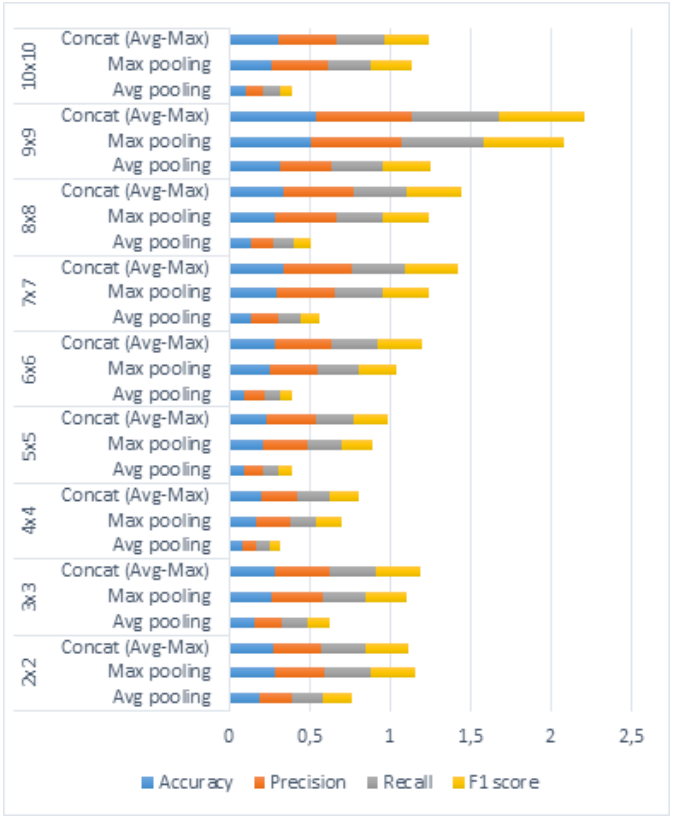

FIGURE 12. Accuracy, precision, recall, and F1 score results of pooling methods with various pool sizes in the ResNet-9 model using the Cifar100 dataset.

5. CONCLUSION

In this study, the success of pooling methods, which play a crucial role in the success of CNNs and are frequently preferred, was compared experimentally for various models, datasets, and pool sizes. In addition, the experimental performance of concatenating maximum and average pooling methods to reduce information loss under similar conditions was examined. The concept underlying this method is to incorporate the strengths of both pooling methods. While maximum pooling is effective at preserving the input's most prominent characteristics, average pooling is effective at capturing the data's general tendencies. It was hypothesized that the final feature map derived by combining the feature maps generated by these two methods would provide a more accurate representation of the input by incorporating both the most prominent features and general trends. The experimental

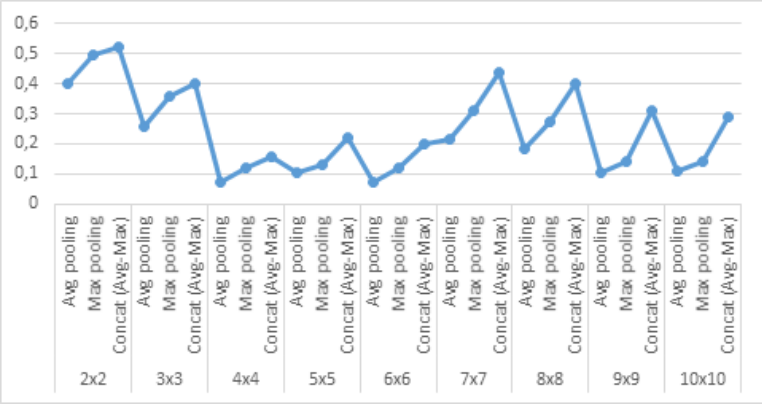

FIGURE 13. F1 score results of pooling methods with various pool sizes in the ResNet-9 model using the SVHN dataset.

results revealed that the superiority of a particular pooling method varied depending on the application scenario. In the shallow model, i.e., LeNet-5, the average pooling outperformed all other methods for small pool sizes across all datasets used in the study. As the pool size increased, the efficacy of the average pooling method deteriorated and fell behind that of the maximum pooling. For large pool sizes, i.e. $> 5x5$, the Concat(Avg, Max) outperformed the other two algorithms. In the ResNet-9 deep model, the max pooling performed better than the other methods for small pool sizes. In this model, the average pooling lagged behind the other methods for all pool sizes. As the size of the pool increased, the Concat(Avg, Max) method provided a more accurate representation and obtained better results. For the SVHN dataset, this method yielded the highest scores for all pool sizes. This study guides selecting pooling methods depending on the model and pool size. The experimental results demonstrated that the pooling method has a significant effect on model performance. Moreover, there were model and pool size-dependent variations among different pooling methods. Future research will investigate the impact of using multiple pooling methods at various levels of deep CNN models.

Declaration of Competing Interests The author declares that there is no competing interest regarding the publication of this paper.

Acknowledgement The numerical calculations reported in this paper were partially performed at TUBITAK ULAKBIM, High Performance and Grid Computing Center (TRUBA resources).

References

[1] Ataş, I., Human gender prediction based on deep transfer learning from panoramic dental radiograph images, *Trait. du Signal*, 39 (5) (2022), 1585, http://dx.doi.org/10.18280/ts.390515.

[2] Ataş, M., Özdemir, C., Ataş, İ., Ak, B., Özeroğlu, E, Biometric identification using panoramic dental radiographic images withfew-shot learning, *Turk. J. Electr. Eng.*, 30 (3) (2022), 1115-1126, http://dx.doi.org/10.55730/1300-0632.3830.

[3] Ozdemir, C., Gedik, M. A., Kaya, Y., Age estimation from left-hand radiographs with deep learning methods, *Trait. du Signal*, 38 (6) (2021), http://dx.doi.org/10.18280/ts.380601.

[4] Krizhevsky, A., Sutskever, I., Hinton, G. E., Imagenet classification with deep convolutional neural networks, *Commun. ACM*, 60 (6) (2017), 84-90, http://dx.doi.org/10.1145/3065386.

[5] Tolstikhin, I. O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Dosovitskiy, A., Mlp-mixer: An all-mlp architecture for vision, *Adv. Neural Inf. Process. Syst.*, 34 (2021), 24261-24272, https://arxiv.org/abs/2105.01601.

[6] Meng, L., Li, H., Chen, B. C., Lan, S., Wu, Z., Jiang, Y. G., Lim, S. N., Adavit: Adaptive vision transformers for efficient image recognition, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (2022), 12309-12318, http://dx.doi.org/10.1109/cvpr52688.2022.01199.

[7] Krizhevsky, A., Nair, V., rey Hinton, G., CIFAR-10 dataset, (2014), Available at: https://www.cs.toronto.edu/ kriz/cifar.html.

[8] Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A., The street view house numbers (SVHN) dataset, (2016). Available at: https://www.kaggle.com/datasets/stanfordu/street-view-house-numbers.

[9] Akhtar, N., Ragavendran, U., Interpretation of intelligence in cnn-pooling processes: a methodological survey, *Neural. Comput. Appl.*, 32 (3) (2020), 879-898, http://dx.doi.org/10.1007/s00521-019-04296-5.

[10] Yu, D., Wang, H., Chen, P., Wei, Z., Mixed pooling for convolutional neural networks, *International Conference on Rough Sets and Knowledge Technology*, (2014), 364-375, http://dx.doi.org/10.1007/978-3-319-11740-9_34.

[11] Dogan Y., A new global pooling method for deep neural networks: Global average of top-k max-pooling, *Trait. du Signal*, 40 (2) (2023), 577-587, http://dx.doi.org/10.18280/ts.400216.

[12] Saeedan, F., Weber, N., Goesele, M., Roth, S., Detail-preserving pooling in deep networks, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2018), 9108-9116, http://dx.doi.org/10.1109/cvpr.2018.00949.

[13] He, K., Zhang, X., Ren, S., Sun, J., Spatial pyramid pooling in deep convolutional networks for visual recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*,37 (9) (2015), 1904-1916, http://dx.doi.org/10.1109/tpami.2015.2389824.

[14] Sun, M., Song, Z., Jiang, X., Pan, J., Pang, Y. Learning pooling for convolutional neural network, *Neurocomputing*, 224, (2017), 96-104, http://dx.doi.org/10.1016/j.neucom.2016.10.049.

[15] Wang, F., Huang, S., Shi, L., Fan, W., The application of series multi-pooling convolutional neural networks for medical image segmentation, *Int. J. Distrib. Sens. Netw.*, 13 (12) (2017), http://dx.doi.org/10.1177/1550147717748899.

[16] Özdemir, C., Avg-topk: A new pooling method for convolutional neural networks, *Expert Syst. Appl.*, (2023), 119892, http://dx.doi.org/10.1016/j.eswa.2023.119892.

[17] Sermanet, P., Chintala, S., LeCun, Y., Convolutional neural networks applied to house numbers digit classification, *Proceedings of the 21st International Conference on Pattern Recognition*, (2012), 3288-3291, https://doi.org/10.48550/arXiv.1204.3968.

[18] Fei, J., Fang, H., Yin, Q., Yang, C., Wang, D., Restricted stochastic pooling for convolutional neural network, *Proceedings of the 10th International Conference on Internet Multimedia Computing and Service*, (2018), 1-4, http://dx.doi.org/10.1145/3240876.3240919.

[19] Wu, H., Gu, X., Max-pooling dropout for regularization of convolutional neural networks, *International Conference on Neural Information Processing*, (2015), 46-54, http://dx.doi.org/10.1007/978-3-319-26532-2_6.

[20] Song, Z., Liu, Y., Song, R., Chen, Z., Yang, J., Zhang, C., Jiang, Q., A sparsitybased stochastic pooling mechanism for deep convolutional neural networks, *Neural Netw.*, 105 (2018), 340-345, http://dx.doi.org/10.1016/j.neunet.2018.05.015.

[21] Tong, Z., Aihara, K., Tanaka, G., A hybrid pooling method for convolutional neural networks, *International Conference on Neural Information Processing*, (2016), 454-461, http://dx.doi.org/10.1007/978-3-319-46672-9_51.

[22] Shahriari, A., Porikli, F., Multipartite pooling for deep convolutional neural networks, arXiv:1710.07435, (2017), http://arxiv.org/abs/1710.07435.

[23] Kumar, A., Ordinal pooling networks: for preserving information over shrinking feature maps, arXiv:1804.02702, (2018), http://arxiv.org/abs/1804.02702.

[24] Kolesnikov, A., Lampert, C. H. Seed, Expand and constrain: three principles for weakly-supervised image segmentation, *European Conference on Computer Vision*, (2016), 695-711, http://dx.doi.org/10.1007/978-3-319-46493-0_42.

[25] Williams, T., Li, R., Wavelet pooling for convolutional neural networks, *International Conference on Learning Representations*, (2018).

[26] Rippel, O., Snoek, J., Adams, R. P., Spectral representations for convolutional neural networks, *Adv. Neural Inf. Process. Syst.*, (2015), 28, https://doi.org/10.48550/arXiv.1506.03767.

[27] Wang, Z., Lan, Q., Huang, D., Wen, M., Combining fft and spectral-pooling for efficient convolution neural network model, *2016 2nd International Conference on Artificial Intelligence and Industrial Engineering (AIIE)*, (2016), 203-206, http://dx.doi.org/10.2991/aiie-16.2016.47.

[28] Simonyan, K., Zisserman, A., Very deep convolutional networks for large-scale image recognition, arXiv:1409.1556, (2014), https://doi.org/10.48550/arXiv.1409.1556.

[29] He, K., Zhang, X., Ren, S., Sun, J., Deep residual learning for image recognition, *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, (2016), 770-778, http://dx.doi.org/10.1109/cvpr.2016.90.

[30] Tan, M., Le, Q., Efficientnet: Rethinking model scaling for convolutional neural networks, *International conference on machine learning (ICML)*, (2019), 6105-6114, https://doi.org/10.48550/arXiv.1905.11946.

[31] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., Gradient-based learning applied to document recognition, *Proc. IEEE*, 86 (1998), 2278–2324, http://dx.doi.org/10.1109/5.726791.

[32] Nair, V., Hinton, G. E., Rectified linear units improve restricted boltzmann machines, *Proceedings of the 27th international conference on machine learning (ICML)*, (2010), 807-814.

[33] Bottou, L., Stochastic gradient descent tricks, Neural Networks: Tricks of the Trade: Second Edition, (2012), 421-436, http://dx.doi.org/10.1007/978-3-642-35289-8_25.

[34] Boureau Y.-L., Le Roux N., Bach F., Ponce J., LeCun Y., Ask the locals: multiway local pooling for image recognition, in Computer Vision, *IEEE International Conference*, (2011), 2651-2658, http://dx.doi.org/10.1109/iccv.2011.6126555.

Appendix

Table 4. Compare pooling methods for Model: Lenet Dataset: SVHN

Pool size	Method	Accuracy	Precision	Recall	F1 score
2x2	Avg pooling	**0.8611**	**0.8483**	**0.8506**	**0.8492**
	Max pooling	0.8436	0.8333	0.8299	0.8305
	Concat (Avg-Max)	0.8557	0.8479	0.8400	0.8427
3x3	Avg pooling	**0.8637**	0.8528	**0.8501**	**0.8508**
	Max pooling	0.8471	0.8351	0.8403	0.8355
	Concat (Avg-Max)	0.8608	**0.8678**	0.8388	0.8496
4x4	Avg pooling	**0.8507**	0.8358	0.8372	0.8352
	Max pooling	0.8256	0.8152	0.8056	0.8087
	Concat (Avg-Max)	0.8486	**0.8361**	**0.8383**	**0.8361**
5x5	Avg pooling	0.8287	0.8098	**0.8114**	0.8080
	Max pooling	0.8007	0.8007	0.7673	0.7784
	Concat (Avg-Max)	**0.8313**	**0.8303**	0.8003	**0.8109**
6x6	Avg pooling	0.8138	0.7923	**0.7925**	0.7892
	Max pooling	0.8024	0.7901	0.7748	0.7804
	Concat (Avg-Max)	**0.8189**	**0.8133**	0.7903	**0.7993**
7x7	Avg pooling	0.7541	0.7390	0.7147	0.7212
	Max pooling	0.7470	0.7485	0.7150	0.7241
	Concat (Avg-Max)	**0.7712**	**0.7608**	**0.7395**	**0.7399**
8x8	Avg pooling	0.7427	0.7144	0.7033	0.6990
	Max pooling	0.7446	0.7277	0.7050	0.7071
	Concat (Avg-Max)	**0.7894**	**0.7870**	**0.7515**	**0.7642**
9x9	Avg pooling	0.6447	0.6016	0.5839	0.5757
	Max pooling	0.7253	0.7085	0.6869	0.6911
	Concat (Avg-Max)	**0.7763**	**0.7635**	**0.7448**	**0.7516**
10x10	Avg pooling	0.3690	0.3239	0.3061	0.2941
	Max pooling	0.7095	0.7064	0.6641	0.6712
	Concat (Avg-Max)	**0.7461**	**0.7369**	**0.7255**	**0.7261**

TABLE 5. Compare pooling methods for Model: ResNet-9 Dataset: Cifar10

Pool size	**Method**	**Accuracy**	**Precision**	**Recall**	**F1 score**
2x2	Avg pooling	0.4689	0.4764	0.4689	0.4646
	Max pooling	**0.5984**	**0.6031**	**0.5984**	**0.5958**
	Concat (Avg-Max)	0.5845	0.6016	0.5845	0.5859
3x3	Avg pooling	0.3825	0.3971	0.3825	0.3730
	Max pooling	0.5272	0.5498	0.5272	0.5221
	Concat (Avg-Max)	**0.5476**	**0.5730**	**0.5476**	**0.5465**
4x4	Avg pooling	0.2870	0.2822	0.2870	0.2584
	Max pooling	0.4014	0.4392	0.4014	0.3812
	Concat (Avg-Max)	**0.4209**	**0.4782**	**0.4209**	**0.4134**
5x5	Avg pooling	0.3118	0.3098	0.3118	0.2904
	Max pooling	0.4439	0.5032	0.4439	0.4290
	Concat (Avg-Max)	**0.5045**	**0.5261**	**0.5045**	**0.5034**
6x6	Avg pooling	0.3266	0.3391	0.3266	0.3154
	Max pooling	0.5034	0.5621	0.5034	0.4923
	Concat (Avg-Max)	**0.5675**	**0.5907**	**0.5675**	**0.5636**
7x7	Avg pooling	0.3507	0.3660	0.3507	0.3411
	Max pooling	0.5193	0.5892	0.5193	0.5093
	Concat (Avg-Max)	**0.6044**	**0.6175**	**0.6044**	**0.6018**
8x8	Avg pooling	0.3749	0.3930	0.3749	0.3668
	Max pooling	0.5352	0.6164	0.5352	0.5264
	Concat (Avg-Max)	**0.6413**	**0.6444**	**0.6413**	**0.6400**
9x9	Avg pooling	0.3009	0.3090	0.3009	0.2805
	Max pooling	0.4882	0.5558	0.4882	0.4778
	Concat (Avg-Max)	**0.5520**	**0.5830**	**0.5520**	**0.5504**
10x10	Avg pooling	0.3173	0.3375	0.3173	0.2982
	Max pooling	0.5357	0.5911	0.5357	0.5291
	Concat (Avg-Max)	**0.5926**	**0.6139**	**0.5926**	**0.5887**

TABLE 6. Compare pooling methods for Model: ResNet-9 Dataset: Cifar100

Pool size	Method	Accuracy	Precision	Recall	F1 score
2x2	Avg pooling	0.1893	0.2055	0.1893	0.1779
	Max pooling	**0.2841**	**0.3104**	**0.2841**	**0.2809**
	Concat (Avg-Max)	0.2714	0.3031	0.2714	0.2659
3x3	Avg pooling	0.1567	0.1701	0.1567	0.1383
	Max pooling	0.2617	0.3237	0.2617	0.2549
	Concat (Avg-Max)	**0.2884**	**0.3369**	**0.2884**	**0.2766**
4x4	Avg pooling	0.082	0.0899	0.082	0.064
	Max pooling	0.1658	0.2103	0.1658	0.1527
	Concat (Avg-Max)	**0.1954**	**0.2315**	**0.1954**	**0.1789**
5x5	Avg pooling	0.097	0.1138	0.0972	0.0783
	Max pooling	0.2050	0.2845	0.2050	0.1929
	Concat (Avg-Max)	**0.2334**	**0.3034**	**0.2334**	**0.2150**
6x6	Avg pooling	0.0937	0.1247	0.0937	0.0767
	Max pooling	0.2476	0.3048	0.2476	0.2350
	Concat (Avg-Max)	**0.2880**	**0.3448**	**0.2880**	**0.2751**
7x7	Avg pooling	0.1383	0.1625	0.1383	0.1223
	Max pooling	0.2916	0.3696	0.2916	0.2873
	Concat (Avg-Max)	**0.3323**	**0.4265**	**0.3323**	**0.3296**
8x8	Avg pooling	0.1305	0.1389	0.1305	0.1111
	Max pooling	0.2886	0.3755	0.2886	0.2867
	Concat (Avg-Max)	**0.3382**	**0.4307**	**0.3382**	**0.3309**
9x9	Avg pooling	0.3186	0.3212	0.3186	0.2969
	Max pooling	0.5047	0.5709	0.5047	0.5032
	Concat (Avg-Max)	**0.5418**	**0.5936**	**0.5418**	**0.5355**
10x10	Avg pooling	0.1001	0.1139	0.1001	0.076
	Max pooling	0.2674	0.3421	0.2674	0.2552
	Concat (Avg-Max)	**0.2997**	**0.3616**	**0.2997**	**0.2840**

TABLE 7. Compare pooling methods for Model: ResNet-9 Dataset: SVHN

Pool size	**Method**	**Accuracy**	**Precision**	**Recall**	**F1 score**
2x2	Avg pooling	0.4552	0.4219	0.4044	0.4014
	Max pooling	0.5512	0.5250	0.4914	0.4959
	Concat (Avg-Max)	**0.5754**	**0.5680**	**0.5104**	**0.5243**
3x3	Avg pooling	0.3539	0.3074	0.2704	0.2591
	Max pooling	0.4305	0.3911	0.3585	0.3559
	Concat (Avg-Max)	**0.4706**	**0.4427**	**0.4016**	**0.3988**
4x4	Avg pooling	0.1903	0.0921	0.1140	0.0745
	Max pooling	0.2392	0.1759	0.1551	0.1200
	Concat (Avg-Max)	**0.2718**	**0.2278**	**0.1833**	**0.1598**
5x5	Avg pooling	0.2004	0.1565	0.1392	0.1066
	Max pooling	0.2741	0.1871	0.1791	0.1292
	Concat (Avg-Max)	**0.3168**	**0.2451**	**0.2358**	**0.2182**
6x6	Avg pooling	0.1979	0.1028	0.1163	0.075
	Max pooling	0.2374	0.1228	0.1645	0.1216
	Concat (Avg-Max)	**0.3178**	**0.2419**	**0.2278**	**0.1982**
7x7	Avg pooling	0.2974	0.2744	0.2307	0.2166
	Max pooling	0.3953	0.3726	0.3170	0.3130
	Concat (Avg-Max)	**0.4897**	**0.4928**	**0.4359**	**0.4364**
8x8	Avg pooling	0.2400	0.2424	0.1983	0.1819
	Max pooling	0.3657	0.3459	0.2862	0.2709
	Concat (Avg-Max)	**0.4671**	**0.4681**	**0.3995**	**0.4028**
9x9	Avg pooling	0.1851	0.1374	0.1426	0.1041
	Max pooling	0.2769	0.1997	0.1856	0.1439
	Concat (Avg-Max)	**0.3929**	**0.3665**	**0.3222**	**0.3093**
10x10	Avg pooling	0.2017	0.1523	0.1477	0.1098
	Max pooling	0.2680	0.2503	0.1823	0.1393
	Concat (Avg-Max)	**0.3754**	**0.3875**	**0.3057**	**0.2871**