İSTANBUL UNIVERSITY CERRAHPAŞA

ORIGINAL ARTICLE
Electrica 2025; 25: 1-13

Improving the Performance of Ensemble Models With Class Activation Maps

Aykut Görkem Gelen

Department of Electrical and Electronics Engineering, Erzincan Binali Yıldırım University Faculty of Engineering, Erzincan, Turkiye

Cite this article as: A. G. Gelen, "Improving the performance of ensemble models with class activation maps," *Electrica*, 25, 0184, 2025. doi: 10.5152/electrica.2025.24184.

WHAT IS ALREADY KNOWN ON THIS TOPIC?

- *Combining the outputs or extracted features of multiple models has been shown to enhance test accuracy by taking advantage of the distinct dynamics of each model. This technique is known as model ensembling.*
- *In deep learning, class activation maps visualize which parts of an image contribute to the model's predictions. Studies have used this technique to analyze the predicted labels and enhance the model's explainability.*

WHAT THIS STUDY ADDS ON THIS TOPIC?

- *This study shows that class activation maps can also guide the combination of multiple models beyond their standard use. The study shows that the best ensemble performance comes from*

Corresponding author:
Aykut Görkem Gelen

E-mail:
aykut.gelen@erzincan.edu.tr

Received: December 2, 2024
Revision Requested: February 20, 2025
Last Revision Received: March 1, 2025
Accepted: March 12, 2025
Publication Date: June 3, 2025

DOI: 10.5152/electrica.2025.24184

ABSTRACT

Convolutional neural networks with different architectures extract different features. Imagine a scenario where multiple models were trained on the same dataset, with the same hyperparameters, and under the same conditions. Because of their architecture, these models will learn to extract various features. As a result, even when all conditions are identical, different architectures will achieve varying levels of test accuracy. Test accuracy can be improved even more through ensembling models with varying levels of accuracy in different ways. The purpose of this research is to examine the encoder layers of models trained in such a scenario and to compare the effects of various architectures. In the study, the CIFAR-100 dataset was used to train five models based on ResNet18, GoogleNet, and MobileNetv3L. These models were ensembled in a multitude of ways. Class activation maps were computed for both analysis and ensembling. The research demonstrated that class activation maps could be used to learn how to ensemble models. As a result, model ensembling using class activation maps significantly improved test accuracy. Furthermore, one of the most important findings from the study is that including a model with low test accuracy in the ensemble can boost overall success.

Index Terms— CIFAR-100, class activation maps, ensemble models

I. INTRODUCTION

Researchers have studied classification problems for a long time. Surprisingly, some datasets containing a large number of labels still exhibit low levels of test accuracy in classification scenarios [1]. For example, the dataset called CIFAR-100 contains very small images, 32×32 pixels in size. Compared to standard models that can classify other datasets, such as ImageNet [2] or Modified National Institute of Standards and Technology (MNIST) [3], with almost absolute accuracy, the classification success of the CIFAR-100 dataset is quite low [4]. This is due to the small resolution of the images, the vast number of classes, and the generalization capabilities of model architectures.

Convolutional neural networks extract certain features that can describe images by convolving learnable filters over the image. These features can be thought of as specific parts of the image, repeating sections, or distinct edges and corners. Later, multi-layer perceptrons (MLPs) classify the transformed images into vectors, a series of features, in a manner similar to classical artificial neural networks. In convolutional networks, the MLP only distinguishes the features extracted by the encoder. Therefore, the encoder part typically bears full responsibility for test accuracy. The features extracted in the encoder part need to be descriptive and generalizable. Otherwise, the MLP might not be able to fully distinguish features into classes. Consequently, any classification error in a dataset indicates a lack of meaningful extraction of information from the encoder part. Furthermore, it is important to remember that trained convolutional networks operate as a black-box system. This implies that each architecture can encode the relationships between the same input and output data in different ways. Therefore, a basic analysis can be conducted to determine which regions in the image correspond to each class. The literature refers to this metric as class activation maps [5]. Class activation maps are the specific regions of the image that the encoder layers focus on when calculating the output for

combining the predictions of multiple models with class activation map-gated learnable linear layers.

- *One of the key findings of the research is that including models with relatively low test accuracy in an ensemble can contribute positively to the overall performance.*
- *The research provides an evaluation of different Convolutional Neural Network (CNN) architectures (ResNet18, GoogleNet, and MobileNetv3L) on the CIFAR-100 dataset, indicating how different architectures might enhance test performance by model ensembling.*

each class. These maps are calculated using an image and a class label. Therefore, maps can be produced using both correct and incorrect classes. Thus, if the model selects the wrong class, the reasons for this misclassification can be investigated.

Each model uses different approaches when developing convolutional neural networks, such as using skip connections, looking at wider areas, or extracting more features with parallel filters. Therefore, the encoder part of each model extracts different features. Suppose multiple models are trained on the same dataset, using the same hyperparameters and loss function. Each of these models will extract the features expected to represent an image in different ways due to their architecture and black box structure. Even if the size of each feature vector is the same, their values will be different. The features of each model will encode some information about the dataset. The question here is which of these features, or which combinations, represent the image in a more separable space.

In this study, five well-known convolutional network models were trained under identical conditions on the CIFAR-100 dataset to answer the questions. The models achieved different levels of test accuracy as expected. Class activation maps for the encoders of the models were calculated and analyzed. The aim of the study is to examine and improve the ensemble behaviors of models trained under the same conditions. Based on this information, five trained models were combined in different ways to form an ensemble. The most significant contribution of the study is to demonstrate that the information in the class activation maps can improve the overall test accuracy of the ensemble approach. To prove this, a comparison was made with methods that do not use class maps, learnable methods, and standard ensemble methods based on voting. Another contribution of the study is demonstrating that even if a model's test accuracy is lower than others, it can still increase the overall test accuracy.

The rest of this article is organized as follows: The subsequent section presents the related work in a comparative manner. The second section provides background information on convolutional networks, standard models, loss functions, class activation maps, and model ensembling. The third section delves into the research on model ensembling and class activation maps. This section provides model architectures, training details, and hyperparameters. In the next section, the results and discussion are presented. The last section contains the conclusion.

II. RELATED WORK

Zeng et al. have published a comprehensive comparison of classifiers using the CIFAR-100 dataset [1]. This study examines and presents the performance of traditional models such as ResNet, GoogleNet, and models by Visual Geometry Group (VGG). The study emphasizes that ResNet and GoogleNet could potentially solve the problem and attain an average accuracy rate of approximately 73%. He et al. have proposed a learning approach based on residuals for convolutional neural networks [6]. The main objective of this study is to examine the degradation problem. As a result, it has been demonstrated that adding residual connections significantly facilitates the optimization of a convolutional network. The ResNet architecture developed using this approach has shown great success on many datasets, including the ImageNet dataset.

Various variations have been proposed for the ResNet architecture. For instance, Yu et al. discuss in their study of dilated residual networks that the use of dilation enhances the model's capacity for generalization without altering its depth [7]. They demonstrate the method's success in the study. The study argues that the dilation operator increases the receptive fields in deep layers, eliminating the need for subsampling. In the study published by Hu et al., squeeze-and-excitation (SE) blocks have been proposed [8]. They argue that more meaningful features can be extracted by utilizing the dependencies between the channels in the image. It has been stated in the study that models like ResNet can be directly replaced with counterpart SE blocks. The study emphasized that the SE blocks significantly improved the quality of the extracted features.

In the famous work published by Szegedy and others, GoogleNet was proposed [9]. They have emphasized that GoogleNet, which uses the Inception blocks, is highly successful in feature extraction. One of the key points emphasized in the article is that GoogleNet allows for an increase in depth and width while maintaining a fixed computational budget [9]. GoogleNet and its more advanced versions achieve highly successful results for classification and object recognition problems.

MobileNets are convolutional network architectures optimized for mobile devices. Howard et al. published a study that presented MobileNetV3, the advanced version of these architectures [10]. MobileNetV3 consists of two models named MobileNetV3-Large and MobileNetV3-Small. The study emphasized that the MobileNet architecture is highly successful in terms of performance and accuracy for classification, object recognition, and image-based applications.

It is important to understand which features the models extract and how they process them. For this purpose, class activation maps have been developed. Zhou et al., in a very famous paper they published, emphasized the relationship between the global average pooling operator and feature maps [5]. This and previous studies explain that some filters in convolutional networks behave selectively, independent of training [11]. However, it is shown that this situation disappears when the last linear layer is added, and this can be corrected with the Global Average Pool operator. In the study, a method is proposed to find which regions in the image are used for discrimination with a single forward pass.

Proposed by Selvaraju et al. in 2017, GradCam generalizes the class activation map concept for all convolutional networks [12]. Using feature maps from the deeper layers of the network, this method converts the activation maps of the classes into a heatmap of the same size as the image. Thus, selecting a correct or incorrect class allows us to visualize the areas of the image under attention. This method enables us to analyze the important regions of the images after training on any dataset. This approach has been used in many studies to analyze datasets and trained architectures, providing significant improvements [13].

The combination of multiple models to achieve higher success is known as model ensembling in literature. Different methods exist to ensemble the outputs of the models or the extracted features. Mohammed et al. published a survey that thoroughly investigated and reported on these methods [14]. Bagging is the process of training multiple models in parallel and combining their results to enhance success. Boosting involves training models sequentially, with each model aiming to rectify the errors of its predecessor. In stacking, the models are trained independently. Then, a single model is trained to combine these models.

Studies in various fields, such as tomographic image classification and malware analysis, demonstrate the idea of combining the outputs or features of models using linear neural network layers. For COVID-19 classification on computed tomography images, the EnsembleDVX model combines the DenseNet169, VGG16, and Xception architectures in an ensemble model [15]. This study uses a neural network to combine the models' outputs, and the three architectures yield the best results. In their study, Khan et al. suggested the SB-BR-STM (Squeezed Boosted Boundary Region Split Transform Merge) method, which uses model ensembling to find malware in IoT devices [16].

In the literature, class activation maps, Grad-CAM, and guided Grad-CAM methods are used for analysis purposes [12]. Especially in studies proposed on tomography images, class activation maps, and model ensembling collaborations are encountered. Hosny et al. used Grad-CAM as the explainable model in their study. Grad-CAM was used in the last convolutional layer of DenseNet121 and InceptionV3 to store spatial information and combine it with high-level semantic data [17]. Similarly, Kumaran et al. trained three models—VGG16,

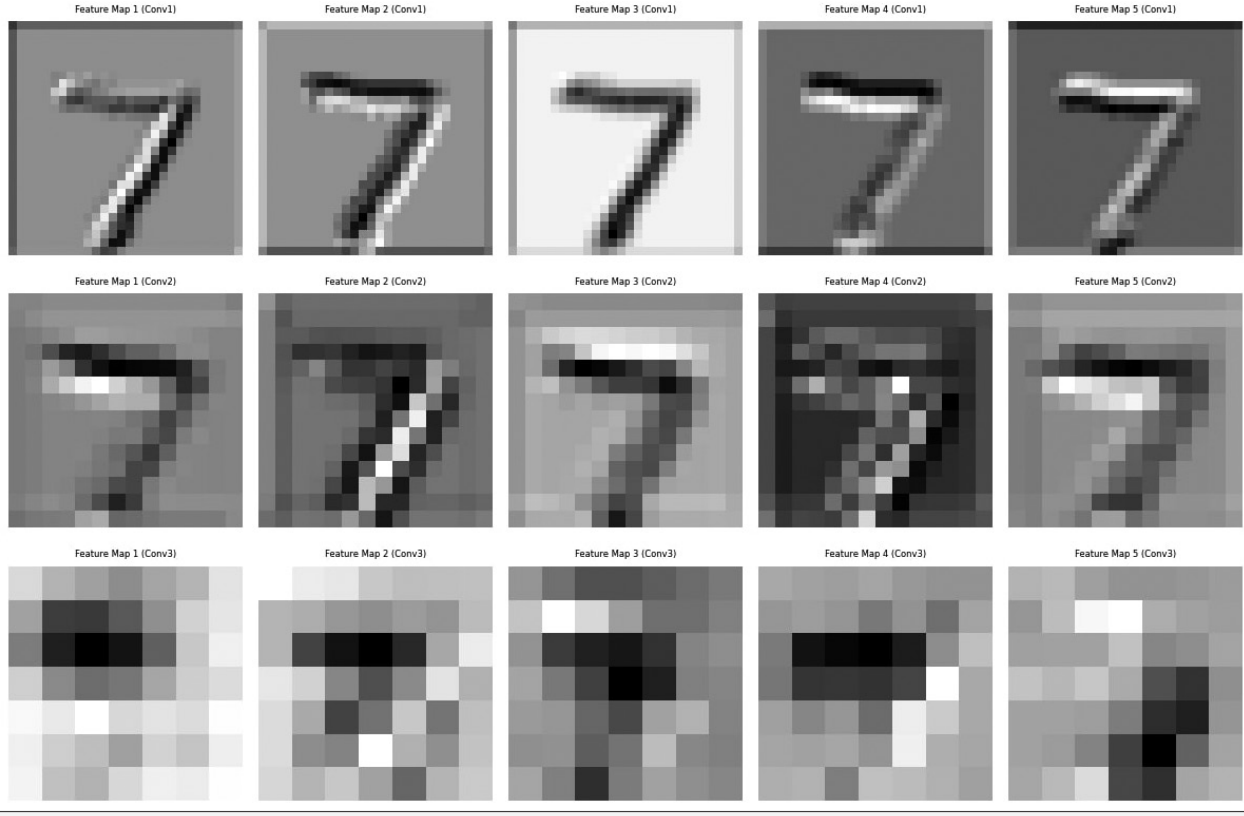

Fig. 1. Feature maps from different layers after training.

ResNet50, and InceptionV3—and attempted to improve tumor detection success through ensembling [18]. These studies use transfer learning and combine the outputs of pre-trained models to obtain results. Additionally, they solely use class activation maps for analysis.

This study is conducted to investigate whether the information from class activation maps can enhance success. In this study, a convolutional network processes the calculated class activation maps into feature vectors, which then serve as a guide for model ensembling. The maps are used in training. All models have been trained with data, and transfer learning has not been used. For these reasons, this study employs a different approach compared to other studies in the literature. Similarly, Birdal demonstrates that incorporating innovative features and optimizing hyperparameters can significantly improve model performance [19]. Both studies show that the innovative use of feature vectors enhances model performance.

III. BACKGROUND

A. Convolutional Neural Networks as Classifiers

The transition from multi-layer perceptrons to convolutional networks is quite important. This great transformation was made possible thanks to many names [20]. These types of networks can create receptive fields like real neural networks thanks to the convolution operator. Thus, processing the data becomes quite easy. For example, processing a large image with an MLP requires layers composed of an infeasible number of neurons. Convolutional networks extract a set of features from this image, and an MLP can perform the same task with fewer neurons by classifying these features. Thus, the problem becomes feasible.

Convolutional networks convolve learnable filters of varying numbers and sizes over the image as convolution kernels. This process results in the production of feature maps. This structure repeats sequentially in layers, and the networks deepen. Fig. 1 shows the feature maps from different layers of a convolutional network that was trained to classify handwritten numbers. As can be seen from the figures, more general features are extracted in the initial layers of the network, while more detailed features are extracted in the deeper layers. Applying convolution causes the image dimensions to decrease or reduce to the desired extent. Thus, the feature maps obtained after processing at certain layers are flattened and converted into a set of features. An MLP network then classifies these features. Any network that undergoes training with a dataset forms feature maps that accurately describe the dataset. Optimization modifies the convolution kernels that provide these transformations.

B. Class Activation Maps

A class activation map (CAM) depicts the discriminative image regions that the CNN uses to identify specific categories [5]. CAMs are utilized to analyze, clarify, and improve convolutional networks. Class activation maps are calculated using the feature maps from the last convolutional layer and the classification weights. This method is limited to the extraction of feature maps only.

GradCAM is a generalized version of CAMs for convolutional networks [12]. Grad-CAM calculates a CAM using the gradients from the last layer of the model. Gradients represent the features that affect the model's output, so identifying the regions that influence decisions can yield much more accurate results. Class activation map generates a localization map for an image classification CNN characterized by an architecture in which global average pooled convolutional feature maps are directly input into the softmax layer [12].

For a target layer producing k feature maps, when calculating GradCAM, the activations A_{ij} at the i,j position are subjected to global average pooling for each class C to produce the score Y^C. Equation (1) provides the calculation of a score Y^C for class C. In the equation, the weights are represented by w_{ij} [12].

$$Y^C = \sum_k w_k^C \frac{1}{Z} \sum_i \sum_j A_{ij}^k \tag{1}$$

Grad-CAM computes the gradient of the class score relative to the feature maps of the convolutional layer. The contribution of the feature maps is assessed by calculating the weight assigned to each feature map by these gradients. These weights illustrate the correlation between each feature map and the class score. Weights α_k^C are calculated over all pixels as shown in Equation (2) [12].

$$\alpha_k^C = \frac{1}{Z} \sum_i \sum_j \frac{\partial Y^C}{\partial A_{ij}^k} \tag{2}$$

C. Ensemble Models

The goal of the Model Ensembling approach is to combine many models to achieve greater accuracy and generalization ability than a single model. By integrating each model's advantages, this approach reduces possible shortcomings. In general, ensemble learning attempts to improve outcomes by merging the outputs or feature maps of several models. One of the most important methods is called Stacking. Stacking is the combination of different models (for example, models that can extract different features). Here, each model is trained separately, and the final model's prediction is made by combining the outputs of these models.

Stacking generally uses a meta-model. The predictions of each submodel are used as input for the meta-model, and the meta-model makes the final predictions. When the submodels are convolutional networks, the meta-model can utilize the feature maps, which are the encoder outputs of the submodels, as inputs. The meta-model learns to combine the features of the submodels in the most effective manner, acting as a black box learning. Thus, different ways of encoding information can yield higher test accuracy results.

The Voting Method is another simple but effective method for model ensembling. This method combines the outputs of multiple models to make a final prediction. It is widely used, particularly in classification problems. Majority voting is the most basic and widely used voting method. In this method, the class predicted by each model is treated as a single "vote," and the class with the most votes is accepted as the final decision. In other words, the class chosen by

TABLE I. MEAN AND SD OF THE CHANNELS IN THE CIFAR-100 DATASET

Channel	Mean	SD
R	0.5070	0.2673
G	0.4865	0.2564
B	0.4409	0.2761

most models is the final prediction. Voting can be easily calculated mathematically by taking the mode of the predictions.

IV. EXPERIMENTS

A group of five models called ResNet-18, SE-ResNet-18, Dilated ResNet-18, MobileNet v3L, and GoogleNet was trained on the CIFAR-100 dataset, and then their accuracy was compared. The model predictions were combined using the voting ensemble method to find a baseline accuracy level. This was conducted to see if the models could be more accurate when used together than when used individually. Later, the models were combined with learnable parameters. Finally, the Grad-CAM gating mechanism, proposed as a better way to combine the models, was used, and its effect was investigated.

A. Dataset and Preprocessing

The CIFAR-100 dataset was used in the study. This dataset is a labeled subset of the 80 million tiny images dataset [21]. The set contains small-resolution images, i.e., 32 × 32 pixels. About 50 000 samples of the cluster are reserved for training and 10 000 samples for testing. Images are color and 3-channel. The 100 classes are divided into 20 subclasses, such as aquatic mammals, food containers, and large man-made outdoor things. Each class in the dataset contains an equal number of 600 examples. The mean and standard deviation of all samples in the training set were calculated to normalize the samples in the dataset. These values are given in Table I.

Using these mean μ and SDs σ, the normalization given by (3) was applied to each channel *x* of the images.

$$z = \frac{(x - ..)}{\sigma} \tag{3}$$

The fact that the images in the dataset are very small in size and contain many classes leaves us faced with a very difficult classification problem. However, using a dataset with multiple classes is extremely beneficial for the conducted research in terms of analyzing and visualizing the model encoder performances. Thus, the effects of model architectures on generalization become apparent. Furthermore, because this dataset is a famous one, it allows for easier benchmarking and comparison. Using data augmentation techniques has been reported to significantly increase the test accuracy compared to using the dataset directly [22]. For this reason, data augmentation techniques were applied in this study. In this way, it was tried to ensure that the models could learn more general features from the dataset. It is also predicted that it will have significant positive effects on reducing overfitting behavior that occurs because of some long training on the dataset. One of the methods used for data augmentation is the random horizontal flipping of the image with a uniform probability distribution. Another method is to randomly crop with a 4-pixel padding. The image can be randomly rotated by a maximum of 15°. Image color properties can be randomly altered (Color Jitter). For this, brightness, contrast, saturation, and hue are adjusted by ±0.2, ±0.2, ±0.2, and ±0.1, respectively. Finally, the image will be normalized with the statistics specified in Table I.

B. Model Architectures

ResNet, GoogleNet, and MobileNet are the architectures frequently used in general classification problems. These architectures are very successful in solving different classification problems [1]. In this study, a total of 5 models were used, including 3 variants based on ResNet architecture (ResNet-18, SE-ResNet-18, Dilated ResNet-18), GoogleNet, and MobileNetv3 Large models. ResNet-18, the smallest model with ResNet architecture, was chosen as the base model for this study. This model allows rapid training with an acceptably low number of parameters. Since it was investigated that encoder structures can learn different features, the D-ResNet-18 variant was created by adding dilations. Dilations were added to enable the convolutional layers in the encoder to scan larger areas and extract more comprehensive features. Additionally, the SE-ResNet-18 variant was trained using SE blocks. Since it is known that SE blocks can learn advanced features, they were used to investigate the differences in feature extraction. In addition, GoogleNet, which uses inception blocks that can extract more advanced and comprehensive features, was chosen as another model. Inception blocks, which can extract different features at various scales with parallel convolution layers, are perfectly compatible with the aim of the research. Finally, MobileNet v3-Large, a model with similar capabilities optimized to reduce the computational load, was chosen to be trained on the same data. An analysis including comparisons of all models is given in Table II.

In the following section, simplified block diagrams of all models are given. It is not possible to show the residual connections of these models in a diagram of this scale. For this reason, the diagrams were drawn to include only sequential blocks. For more detailed architectures of the models used in the study, you can review the code in the repository or the model summaries in the architectures folder. A simplified block diagram of the Resnet18 model is given in Fig. 2. The last layer was changed as in the block diagram to standardize the fully connected prediction layers of all the models to be used in this study.

ResNet architecture consists of multiple consecutive convolutions, normalization, and activation layers. It has been shown that the residual connections in this architecture enable the network to be trained more efficiently and solve the vanishing gradient problem

TABLE II. KEY FEATURES AND NUMBER OF PARAMETERS FOR VARIOUS MODELS

Model	No. of Parameters	Key Features
Resnet18 [6]	11 333 540	Residual connections are available, a structure used for general deep learning.
D-Resnet18 [7]	11 333 540	Dilated convolutions are available, enabling feature learning in larger image regions.
SE-Resnet18 [8]	11 420 580	Squeeze-excitation allows the model to focus on more important features within the channel.
GoogleNet [9]	5 888 004	The Inception block allows it to learn features at various scales.
MobileNet v3Large [10]	3 243 668	Depthwise convolutions offer a fast and efficient model optimized for small models.

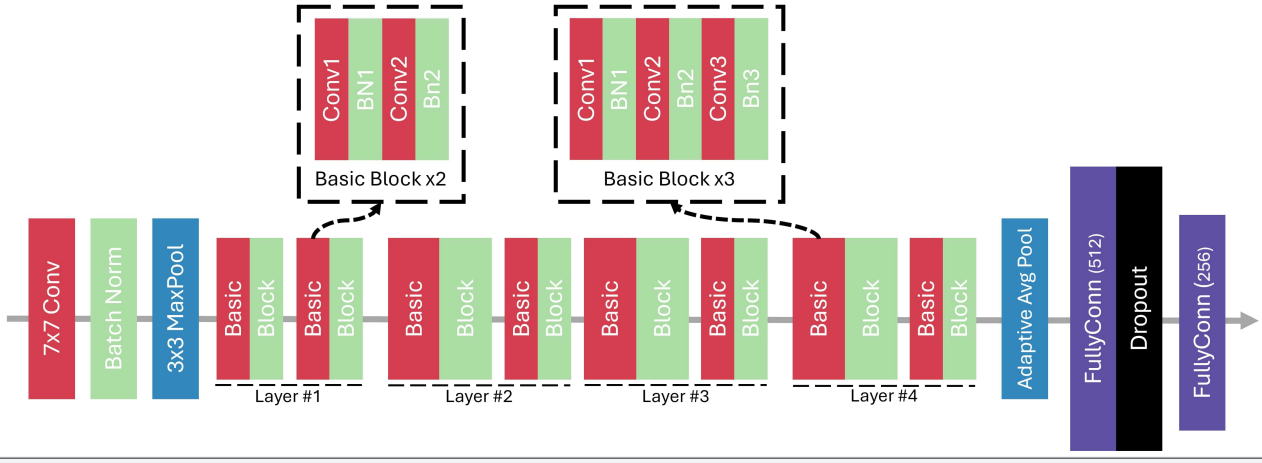

Fig. 2. Block diagram for ResNet18 architecture.

[6]. In the model, the layers up to the adaptive average pool, including the pooling layer, can be called the encoder. The encoder creates a 512-dimensional vector by extracting some key features in the images. The remaining multi-layer perceptron tries to predict classes from these extracted features. Convolutional neural networks frequently use dilation as a feature extraction trick. Filters are expanded with spaces so small-sized filters can scan larger areas and extract more detailed features. To investigate the effect of this feature, the D-ResNet18 variant was created by adding dilation to the ResNet18 model. The block diagram of the D-ResNet18 model is given in Fig. 3. Dilations were added only to layers 3 and 4 to improve deeper features [7]. The dilated convolutional blocks and spacing between kernel points are shown in the detailed block on the block diagram.

The SE block is a structure that uses channel-wise convolutions and tries to use channel information to improve feature extraction [8]. It has been predicted that the SE-ResNet18 variant, created by adding SE blocks to the ResNet18 model, can extract distinctive features by enhancing spatial encoding. The SE-ResNet18 model was created by adding SE blocks to the convolutional blocks of the ResNet18 model and is shown in Fig. 4.

The MobileNet is an architecture optimized for performance. It tries to use fewer computational resources with a structure based on Depthwise Separable Convolutions and SE blocks [10]. MobileNet v3 Large is one of the two models chosen in this study to compare with ResNet models. It has been trained to explain the impact of a different architecture on the extracted features and overall test performance. The architecture of the MobileNet v3 L model is presented in the block diagram given in Fig. 5.

GoogleNet is an important model within convolutional networks that has an inception block architecture. Inception is a block that processes in parallel with kernels of varied sizes simultaneously [9]. Thus, it can learn features of different sizes from the image at once. A block diagram of the Inception block is provided in Fig. 6. The usage of the 1×1 convolutions in this model significantly reduces the number of parameters. GoogleNet is an incredibly famous deep learning model created by sequentially combining inception blocks. It is significantly successful in feature extraction and maintains computational power as the model depth increases [9]. The block diagram of the model is given in Fig. 7. The features extracted by the models are 512-dimensional vectors for ResNet variants, 960-dimensional vectors for MobileNet,

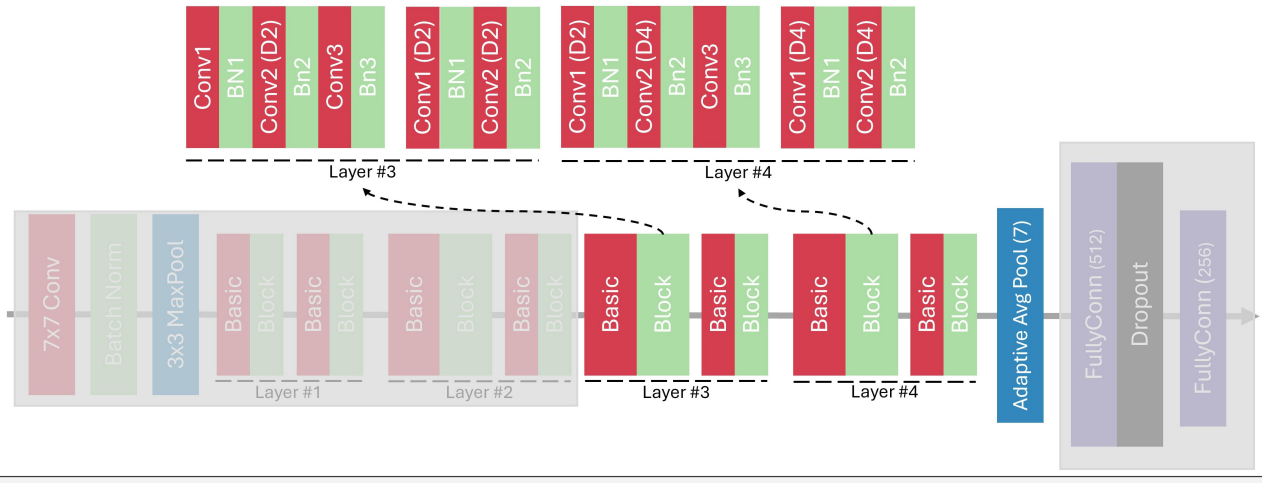

Fig. 3. Block diagram for dilated ResNet18 architecture.

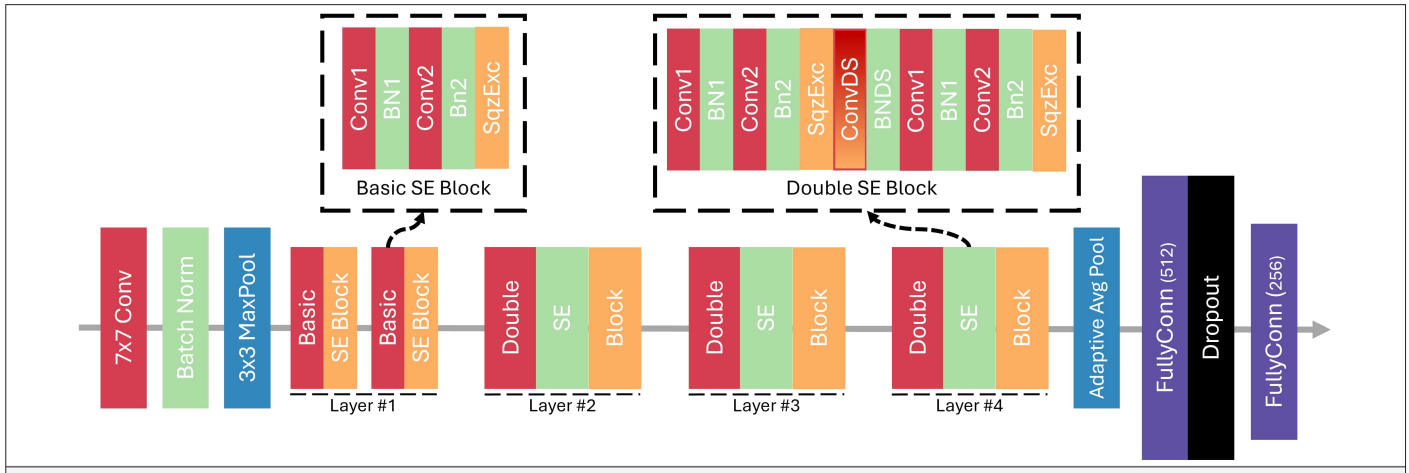

Fig. 4. Block diagram for squeeze and excitation ResNet18 architecture.

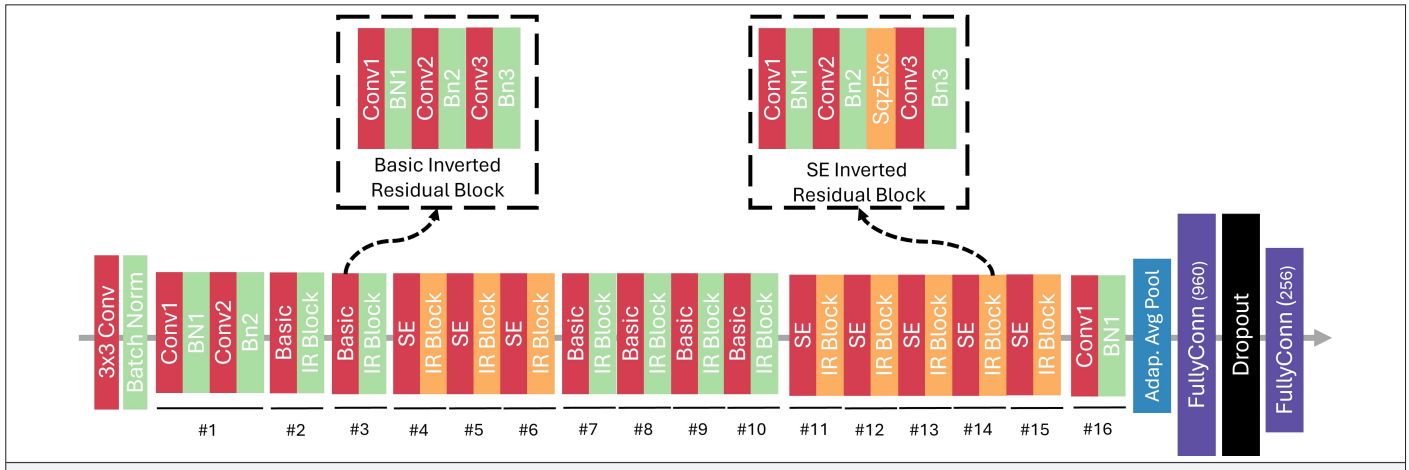

Fig. 5. Block diagram for MobileNet v3 Large architecture.

and 1024-dimensional vectors for GoogleNet. To better position the research, model ensembling was carried out using two approaches. In the first of these approaches, model predictions were combined through voting. In the second approach, three different methods were tried to combine the models with a learnable structure. These are respectively the combination of feature vectors, the direct combination of output vectors, and the combination weighted by GradCam. These methods are explained in

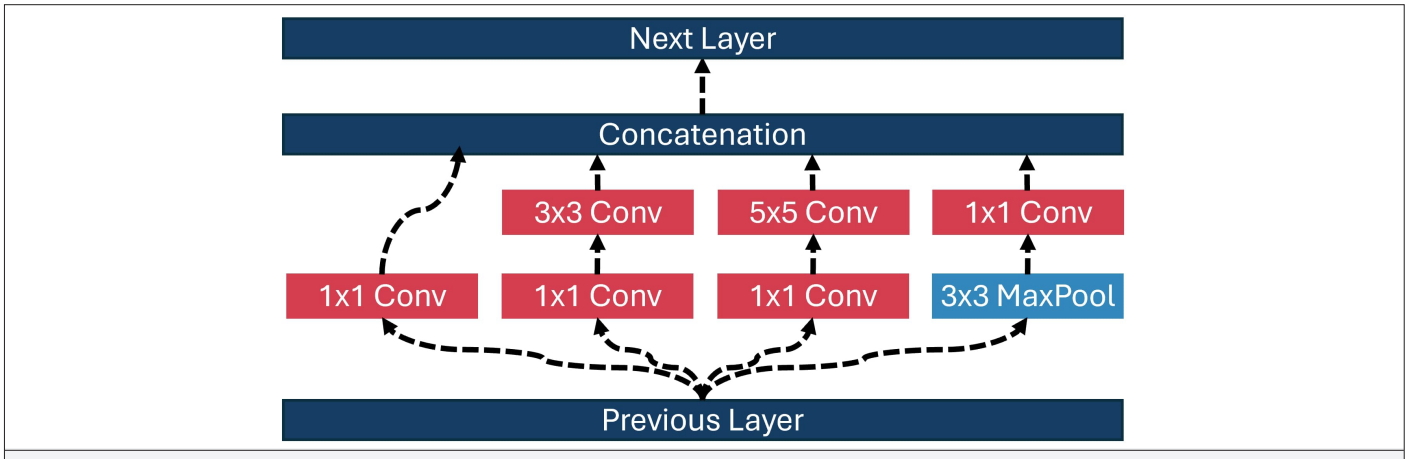

Fig. 6. Block diagram for inception block architecture.

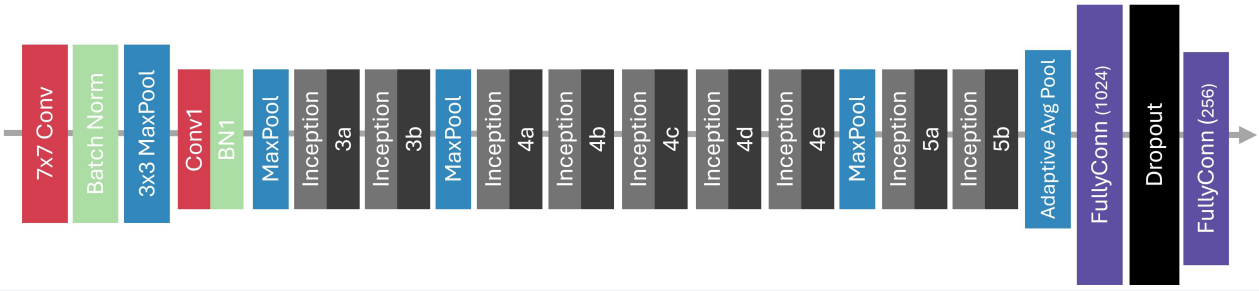

Fig. 7. Block diagram for GoogleNet architecture.

TABLE III. DETAILS FOR ENSEMBLE METHODS

Method	Details
Voting	The modes of the predicted labels were calculated, and direct voting was applied.
Feature concatenation or sum	Linear layers converted each feature vector into 128-dimensional compressed vectors. The vectors are concatenated or summed, classes are predicted.
Output concatenation or sum	Linear layers converted each output vector into 100-dimensional new vectors; vectors are concatenated or summed, classes are predicted.
GradCAM weighted output sum	A quite small convolutional network converts the GradCAM of each sample into 5 vectors of 100 dimensions. These vectors are used to weight the outputs of the models, each of which is 100-dimensional. The resultant vector is calculated as the sum of the products.

Table III. The mentioned methods have been visualized with the block diagrams provided in Fig. 8.

C. Training Setup

In this study, different features that can be learned from the same data are being investigated. Therefore, the training process has been standardized for all models, and hyperparameters have been kept constant across all experiments. In the training of the models, the maximum number of epochs was chosen as 100. It has been observed that the test success does not change after a certain period of training. Therefore, an early stopping strategy has been used. In this strategy, if the test success does not improve in the last n epochs, the training is terminated. In this study, the patience count n has been set to 5 epochs. Therefore, each model has been trained over different epochs.

The stochastic gradient descent algorithm was preferred to optimize the model's weights. The learning rate was set to 0.01, momentum to 0.9, and weight decay to 5e-4. Additionally, the Cosine Annealing Learning Rate Scheduler was used to reduce the learning rate over time. The maximum epoch value for this scheduler was set to 60. Finally, the CrossEntropyLoss function was used to calculate the loss between the model's predictions and the actual labels. To prevent excessively large gradients during the training process, gradient clipping was also applied, and the maximum gradient value was set to 0.1. For ResNet variants and other models, the input images should be 224 × 224 pixels in size. To avoid interfering with the architectures of the models, each example in the CIFAR-100 dataset was bilinearly upscaled by a factor of 7. All training and test data were divided into batches of size 64.

Standard metrics have been used to evaluate the model's success. By calculating the ratio of correct predictions to the total number of predictions, the training and test accuracy for each model have been determined. Additionally, a confusion matrix, which shows

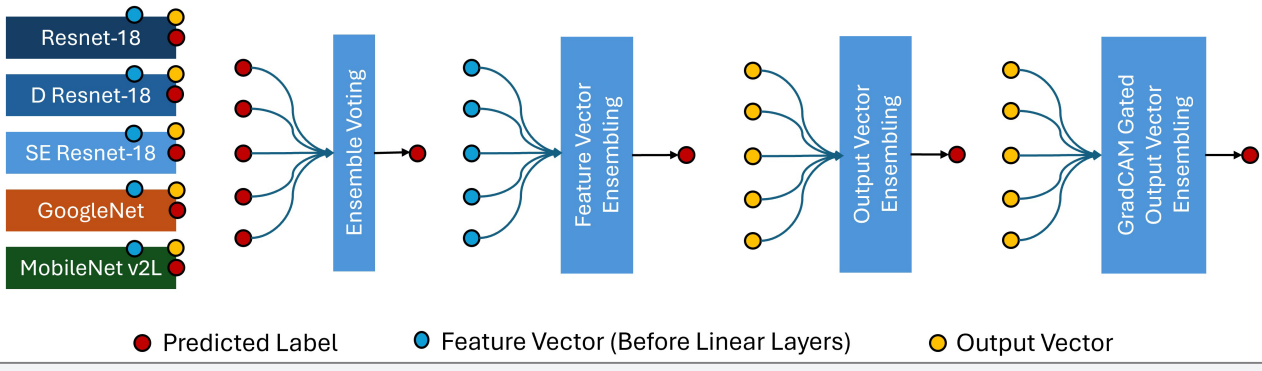

Fig. 8. Block diagram for ensemble models.

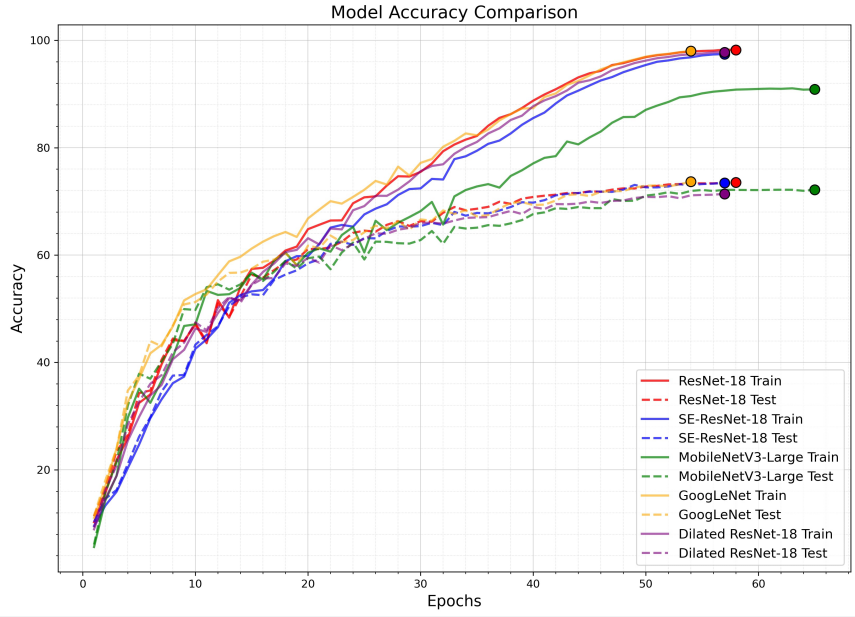

Fig. 9. Train and test accuracy plots for all models.

the correct and incorrect classifications for each class of the model, has been calculated. Standard metrics such as precision, recall, and F1-score have also been calculated for evaluation purposes.

The numerical calculations reported in this paper were fully performed at TUBITAK ULAKBIM, High Performance and Grid Computing Center (TRUBA resources). The computations were carried out on V100 GPUs, and the code was implemented using PyTorch.

V. RESULTS AND DISCUSSION

In this section, accuracy plots, accuracy metrics, and test results have been presented. In this study, five different models were trained under identical conditions. Each of the models stopped training at different epochs due to the training setup, using a specified early stopping strategy. For a fair comparison, the test and training accuracies of the models have been plotted on top of each other and are presented in Fig. 9.

Despite being trained in the same manner, the models have learned different features due to their architectures and have achieved varying levels of accuracy in classification tests. From the accuracy curves of the models, it is observed that the generalization success of all models is at an average level. This situation is close to the levels of the standard models previously reported [1]. Additionally, due to the difficulty of the dataset, the difference between training and test accuracies increases as the number of epochs increases, and the models begin to exhibit overfitting behavior. The accuracy statistics and confusion matrix plots of the models are provided in Table IV and Fig. 10.

When the metrics are examined, it is observed that all models reach the same accuracy levels on average. This suggests that standard classifier convolutional networks have a limit on the CIFAR-100 dataset. Generalizing this idea, it can be said that even if different encoder models are used in the datasets, the average maximum accuracy and generalization capability plateau at a certain level. Additionally, it is observed that the dilation operator among the metrics does not have a positive impact. Fig. 10 shows that the models do not create any bias between classes.

For each model, the same nine randomly selected images belonging to the apple class were used to calculate the class activation maps, which are presented in Fig. 11. The figure displays the class activation maps as heatmaps over the input images. The red areas in the heatmap indicate the places where the model focuses more while making classifications. Examining the figure reveals that

TABLE IV. MODEL PERFORMANCE COMPARISON

Model	Precision	Recall	Specificity	F1 Score	Accuracy (%)
Resnet18	0.7398	0.7355	0.9973	0.7360	73.55
D-Resnet18	0.7174	0.7139	0.9971	0.7143	71.39
SE-Resnet18	0.7369	0.7344	0.9973	0.7345	73.44
GoogleNet	0.7398	0.7369	0.9973	0.7371	73.69
MobileNet v3Large	0.7246	0.7218	0.9971	0.7215	72.18

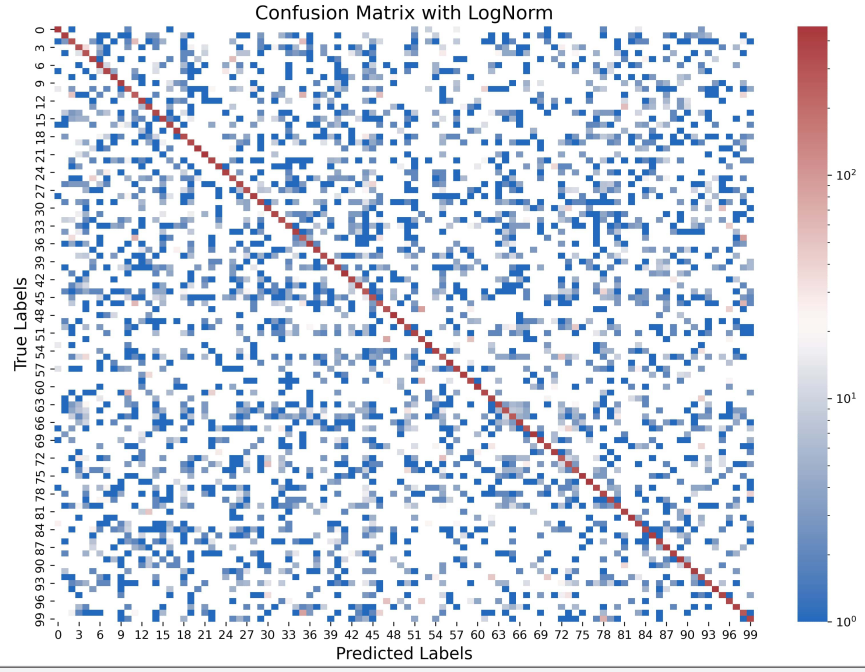

Fig. 10. Summed confusion matrix of all models.

other encoder layers, aside from the dilation operator, draw conclusions from similar regions. The dilated operator, by its nature, focuses on sparse regions and attempts to increase the chance of feature extraction. In the following section, it has been demonstrated that these activation maps can enhance the ensembling performance.

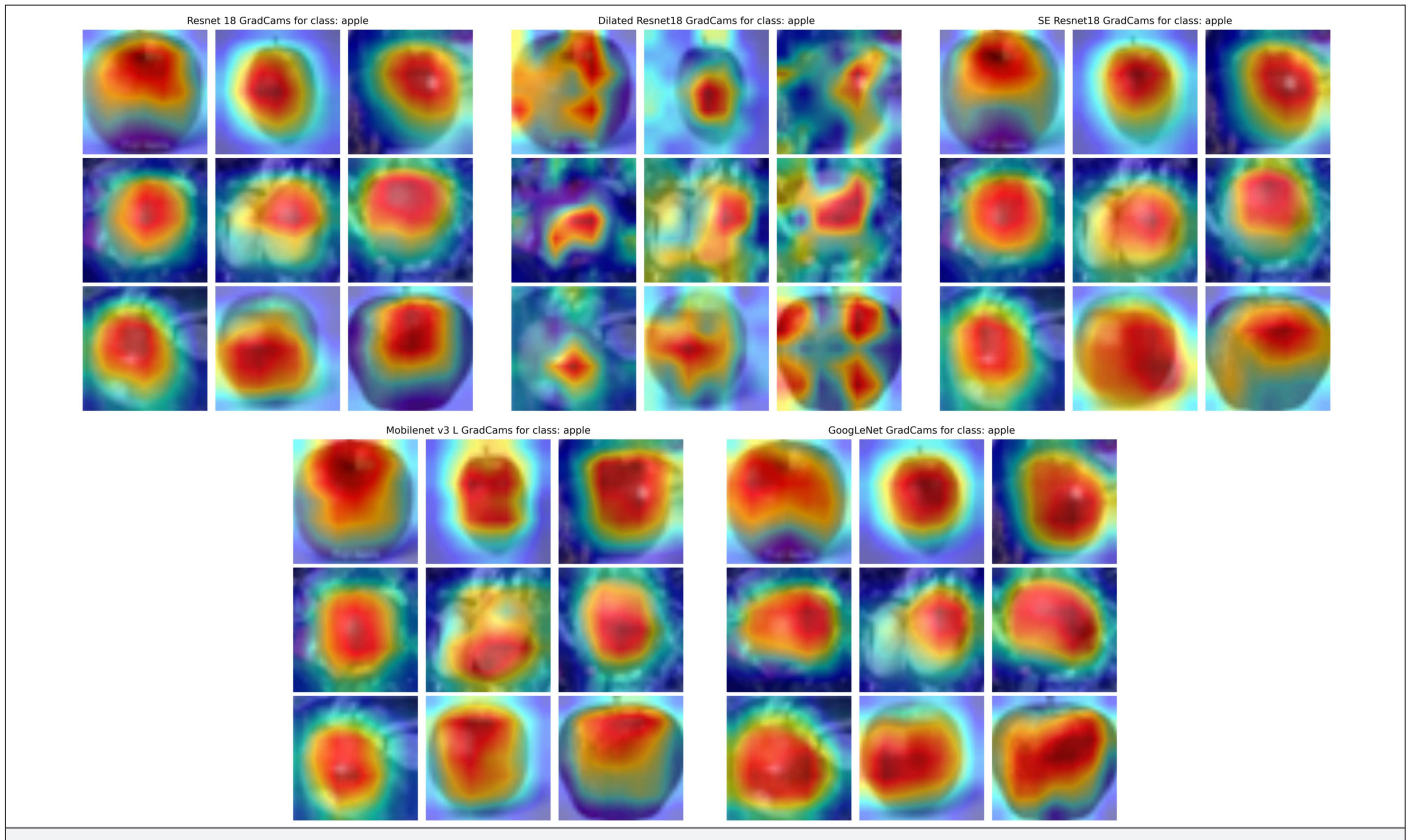

Fig. 11. Class activation maps for random images from the apple class.

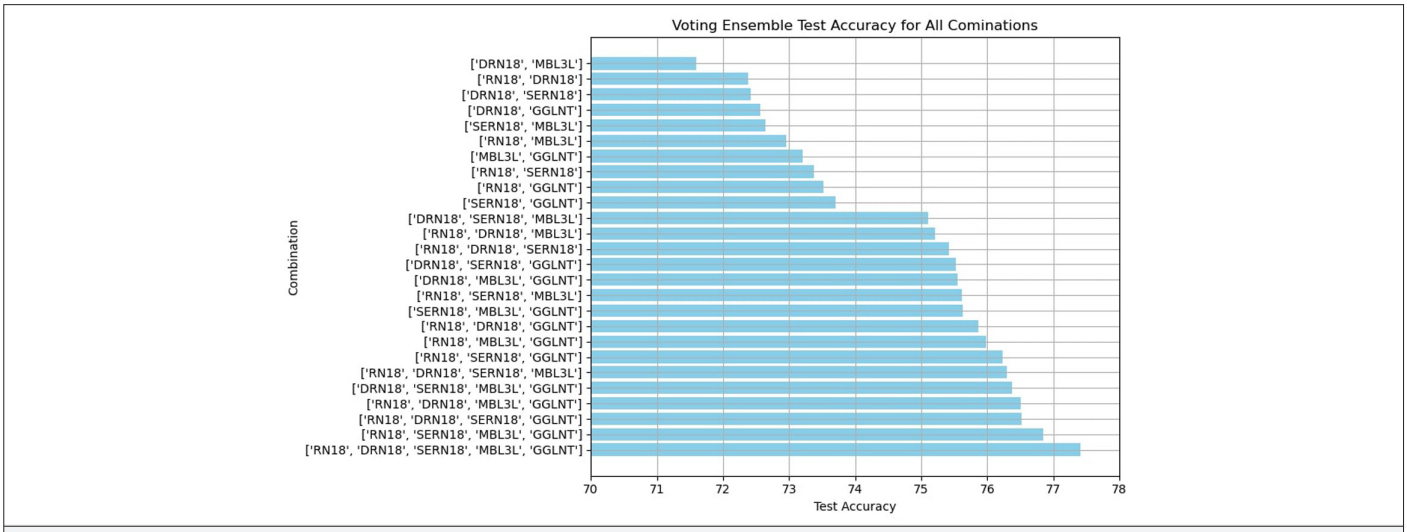

Fig. 12. Test accuracies after voting ensemble.

To investigate whether different models can achieve greater test accuracy through ensembling their outputs and feature vectors, the models have been ensembled. Firstly, the labels predicted by the models were ensembled using voting. To examine the impact of the models that joined the ensemble, the test accuracies after voting for all combinations of all models were calculated and presented as Fig. 12. Looking at the figure, it can be said that the number of models added increases the performance of the voting ensemble. Additionally, the SE ResNet18 and GoogleNet ensembles achieved the best performance in the binary combinations.

Examining Table IV reveals that these two models do not achieve the highest success among the five models. The highest-performing model and the second highest-performing model, the GoogleNet and ResNet18 ensemble, have shown less performance than the previous pair. The Dilated ResNet-18 model, despite its significantly lower performance compared to the others, significantly boosts the success of the ensemble it joins.

Table V displays the results of the second approach, which combines the models with learnable linear or convolutional layers. Since similar results were obtained, only the test accuracies for the scenario where the features or outputs of all models were combined are provided in this section.

Combining the labels predicted by the models through a voting ensemble increases test accuracy by approximately 4% compared to the models' test accuracy. To prove the existence of a strategy that can learn better than the voting ensemble, ensemble results were obtained using the feature vectors or prediction vectors of the models. The predicted feature or output vectors necessitated a gating mechanism for ensembling, as the direct combination was not possible in a trainable scenario. Research demonstrates that this mechanism, utilizing data from GradCAMs, produces superior outcomes compared to voting. Research generally demonstrates that using the predicted vectors yields greater success than using the feature vectors.

TABLE V. TEST ACCURACIES FOR ENSEMBLE METHODS

Method	Test Accuracy (%)
Voting	77.41
Feature concatenation	76.96
Feature sum	76.52
Output concatenation	72.05
Output sum	76.75
GradCAM weighted output sum	78.84

VI. CONCLUSION

This study demonstrated the use of class activation map information to enhance model ensembling. For this purpose, the test accuracies of five different models trained with the same dataset and the same hyperparameters were calculated. The features extracted by the models were examined using class activation maps. As expected, despite all conditions being the same, different models extracted different features and achieved different test accuracies. By freezing the weights of the models, different model combinations were ensembled using voting. The voting results were accepted as a baseline test accuracy value that needed to be surpassed. A small convolutional network processed the class activation maps, transforming the feature maps to weight the class outputs. Then, these weights were used to perform a weighted sum of the models' predictions, significantly increasing the test accuracy. To prove that this increase was due to the information in the class activation maps, test accuracies were also calculated for other learnable configurations that did not include this information. As a result, the scenario using the information from the class activation maps achieved the highest test accuracy.

The experimental results revealed that the scenario using all models yielded the most accurate results. The article's proposed approach achieved the highest test accuracy, resulting in an approximately 2% increase over the voting method. Additionally, despite the model with dilated convolutions having lower test success than the others, its use significantly increased test success in the ensembles. In conclusion, it can be said that models that are more successful than those in the ensemble are not needed to improve overall test

success. Additionally, the study showed that combining the predictions of models provided higher test success compared to combining feature maps.

The proposed method offers opportunities for model ensembling. The research demonstrated the use of GradCAM, an analysis-only method, for feature extraction during forward model operation. Beyond voting, other ensembling mechanisms, such as stacking or boosting, can also be explored as future work. When used in conjunction with CAM-based weighting, these techniques have the potential to significantly enhance model performance.

APPENDIX

The source code for experiments in this research is available at the following GitHub repository: github.com/aggelen/GradCAMGuidedEnsembleModels

The code includes implementations of the experiments and data processing methods described in this paper. You can also access detailed architectures of the models.

Data Availability Statement: The data that support the findings of this study are available on request from the corresponding author.

Peer-review: Externally peer-reviewed.

Author Contributions: Concept – A.G.; Design – A.G.; Supervision – A.G.; Resources – A.G.; Materials – A.G.; Data Collection and/or Processing – A.G.; Analysis and/or Interpretation – A.G.; Literature Search – A.G.; Writing – A.G.; Critical Review – A.G.

Declaration of Interests: The author have no conflicts of interest to declare.

Funding: The authors declare that this study received no financial support.

REFERENCES

1. Y. Zheng, H. Huang, and J. Chen, "Comparative analysis of various models for image classification on Cifar-100 dataset," *J. Phys. Conf. Ser.*, vol. 2711, no. 1, p. 012015, 2024. [CrossRef]
2. J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, 2009, pp. 248–255. [CrossRef]
3. L. Deng, "The MNIST database of handwritten digit images for machine learning research [best of the web]," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 141–142, 2012. [CrossRef]
4. N. Sharma, V. Jain, and A. Mishra, "An analysis of convolutional neural networks for image classification," *Procedia Comput. Sci.*, vol. 132, pp. 377–384, 2018. [CrossRef]
5. B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 2921–2929. [CrossRef]
6. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 770–778. [CrossRef]
7. F. Yu, V. Koltun, and T. Funkhouser, "Dilated residual networks," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 636–644. [CrossRef]
8. J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT: IEEE, pp. 7132–7141, 2018. [CrossRef]
9. C. Szegedy *et al.*, "Going deeper with convolutions," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015, pp. 1–9. [CrossRef]
10. A. Howard *et al.*, "Searching for MobileNetV3," in IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South), 2019, pp. 1314–1324. [CrossRef]
11. B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Object detectors emerge in deep scene CNNs," 2015, *arXiv*. Available: [CrossRef]. [Accessed: Oct. 28, 2024]. [Online]. Available: http://arxiv.org/abs/1412.6856.
12. R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in IEEE International Conference on Computer Vision (ICCV), Venice, 2017, pp. 618–626. [CrossRef]
13. Z. Zhang, Z. Wang, and I. Joe, "CAM-NAS: An efficient and interpretable neural architecture search model based on class activation mapping," *Appl. Sci.*, vol. 13, no. 17, p. 9686, 2023. [CrossRef]
14. A. Mohammed, and R. Kora, "A comprehensive review on ensemble deep learning: Opportunities and challenges," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 35, no. 2, pp. 757–774, 2023. [CrossRef]
15. L. F. de J. Silva, O. A. C. Cortes, and J. O. B. Diniz, "A novel ensemble CNN model for COVID-19 classification in computerized tomography scans," *Results Control. Optim.*, vol. 11, p. 100215, 2023. [CrossRef]
16. S. H. Khan *et al.*, "A new deep boosted CNN and ensemble learning based IoT malware detection," *Comput. Secur.*, vol. 133, p. 103385, 2023. [CrossRef]
17. K. M. Hosny, M. A. Mohammed, R. A. Salama, and A. M. Elshewey, "Explainable ensemble deep learning-based model for brain tumor detection and classification," *Neural Comput. Appl.*, vol. 37, no. 3, pp. 1289–1306, 2025. [CrossRef]
18. Y. Kumaran, J. J. Jeya, M. T. R, S. B. Khan, S. Alzahrani, and M. Alojail, "Explainable lung cancer classification with ensemble transfer learning of VGG16, Resnet50 and InceptionV3 using grad-cam," *BMC Med. Imaging*, vol. 24, no. 1, p. 176, 2024. [CrossRef]
19. R. G. Birdal, "Air pollution impact on forecasting electricity demand utilizing CNN-PSO hyper-parameter optimization," *Environ. Res. Commun.*, vol. 6, no. 5, p. 055022, 2024. [CrossRef]
20. J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, 2015. [CrossRef]
21. A. Krizhevsky, *Learning Multiple Layers of Features from Tiny Images*. University Toronto, Technical Report, 2009.
22. A. Mumuni, and F. Mumuni, "Data augmentation: A comprehensive survey of modern approaches," *Array*, vol. 16, p. 100258, 2022. [CrossRef]

Aykut Görkem GELEN received the M.Sc. degree in electrical and electronics engineering from Karadeniz Technical University, Trabzon, Turkey, in 2017. He received the Ph.D. degree in electrical and electronics engineering from the same university in 2023. He is currently an assistant professor with the Department of Electrical and Electronics Engineering at Erzincan Binali Yıldırım University, Erzincan, Turkey. His research interests include event-based cameras, autonomous systems, spiking neural networks, and convolutional neural networks.