# Ablation study: Achieving 93.87% accuracy in CIFAR10

Miralles Ferrer, Francisco

## 1 Introduction

This study conducts an ablation analysis on the CIFAR-10 test dataset to examine the effect of various architectural and training components on a reference model. Both the individual contribution of each component and their combined influence on overall performance are evaluated. The objective is to reach at least 92% classification accuracy on the CIFAR-10 test split.

The CIFAR-10 dataset consists of color images belonging to ten distinct object categories, including both animals and everyday items. Each sample has a spatial resolution of 32×32 pixels and is assigned to one of ten possible labels. The dataset is partitioned into 50,000 images for training and 10,000 images for evaluation. Representative samples from CIFAR-10 are illustrated in **Figure 1**.

Figure 1: CIFAR10 Example

## 2 Baseline

The starting point is a baseline with the following characteristics. It does not use data augmentation preprocessing techniques, it only normalizes the RGB channel values to [0,1]. It employs a five-layer architecture, each layer consisting of a convolution, a batch normalization, a ReLu activation function, and a max pooling. Feature size increases with each convolution, progressing from 3-32-64-128-256-512. Finally, a linear layer with a ReLU is applied, followed by a final linear layer that projects the features to ten, the number of classes in the dataset.

For training, it uses cross-entropy loss and a stochastic gradient descent optimizer with a fixed learning rate of 0.01, weight loss of 1e-6, and momentum of 0.9.

## 3 Added components

To improve the baseline model's performance, modifications have been made to three fundamental aspects. Different architectures, data augmentation techniques, and training parameters have been proposed.

The impact of each of these modifications is analyzed individually, and subsequently the combination of the best configurations is analyzed through an ablation study, which allows evaluating their specific contribution to the final performance in the test set.

### 3.1 ResNet-18

As a component of the study, a ResNet-18 architecture was used. This model is characterized by the use of residual connections between convolutional blocks, preventing gradient degradation in deep networks. It is represented in **Figure** 2.

The network begins with a first convolutional layer that projects the 3 input channels onto 64 feature maps, followed by batch normalization. Next, four layers are applied, each composed of two residual blocks. While the first stage maintains the resolution and the 64 channels, the following three stages halve the spatial resolution and progressively double the number of channels to reach 512. Finally, a Global Average Pooling layer and a linear layer are applied, projecting the final 512 features onto the 10 classes of the dataset.

Each of the residual blocks consists of a first convolution that converts the number of input features of the layer into the number of output features. Subsequently, it applies batch normalization and ReLU. A second convolution is then added, maintaining the number of channels and another batch normalization. Finally, a residual connection to the block's input and one last ReLU is implemented.

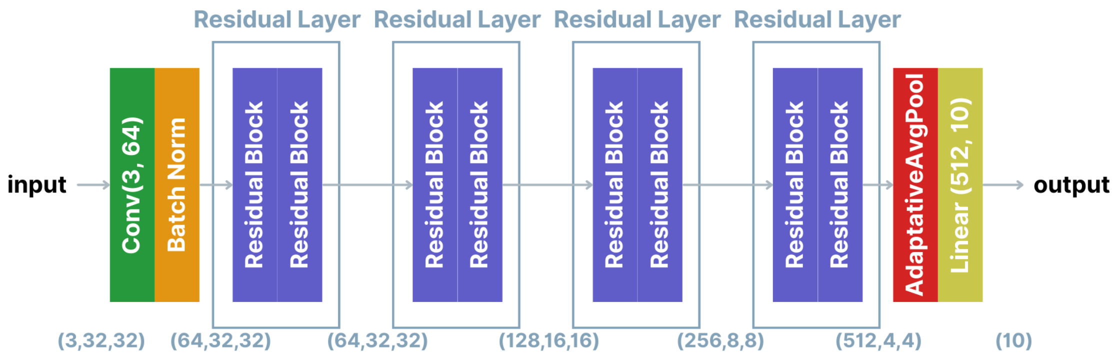

Figure 2: Architecture of the ResNet-18

### 3.2 WideResNet-32

Another ResNet with more convolutions than the previous one has been proposed. ResNet-32, less deep but wider than the standard, has been used to maintain fewer layers but with more channels per block. This model has been called WideResNet-32.

The architecture, represented in **Figure** 3, consists of an initial convolution that projects the image onto a 32-channel space and then applies batch normalization. Subsequently, the network is organized into three main layers, each composed of five residual blocks. The first stage maintains the original resolution and the 32 channels. The following two stages halve the spatial resolution and double the number of channels in their first block, resulting in 128 final channels. Finally, after global average pooling, a linear layer connects the final channels to the 10 output neurons corresponding to the dataset classes.

The total depth of 32 weighted layers is derived from the initial convolution, the 15 residual blocks (each with 2 convolutional layers), and the linear classification layer ($1 + 15 \times 2 + 1 = 32$).

The residual blocks used in this WideResNet-32 have the same architecture as those used in ResNet-18.

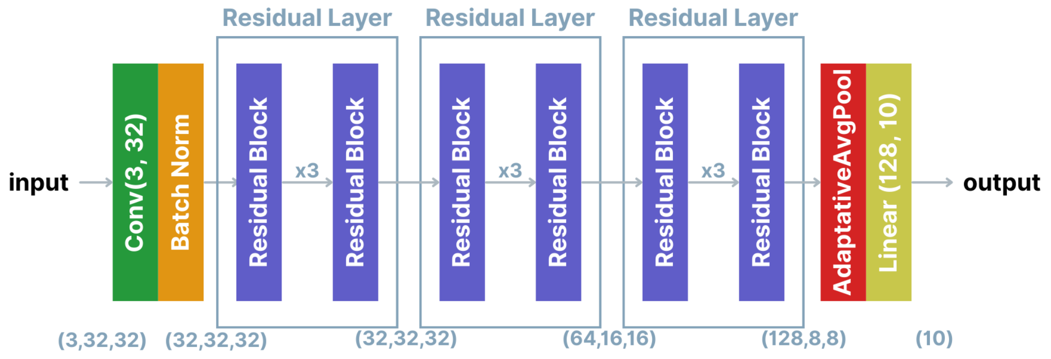

Figure 3: Architecture of the WideResNet-32

## 3.3 DenseNet-100

A DenseNet-100 model, represented in **Figure** 4, has been implemented. This model first applies a convolution to the input image in three channels, increasing to 24 channels. Subsequently, a sequence of three dense blocks interconnected by transition layers is applied.

Each dense block is composed of 16 dense layers, which receive as input all the concatenated outputs of their predecessors within the same block. This connectivity improves gradient flow and enables highly efficient feature reuse. Each of these dense layers adds 12 new feature maps and consists of an initial batch normalization, followed by a ReLU and a convolution that projects the input onto those 12 features.

The transition layers that interconnect the dense blocks consist of batch normalization, ReLU, convolution that reduces the number of channels, and an average pool that reduces the spatial resolution.

After the last of the dense blocks, the architecture has a batch normalization, followed by a ReLU, an average pool, and a linear layer that captures the 342 channels and projects them into 10 output channels, one for each class in the dataset.

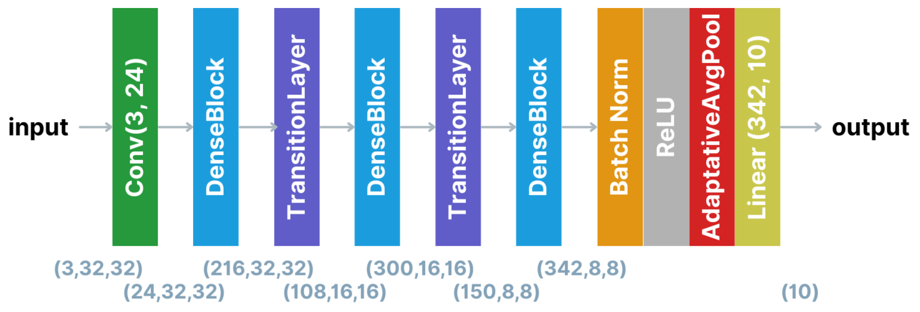

Figure 4: Architecture of the DenseNet-100

## 3.4 Data augmentation

An image preprocessing pipeline has been proposed that includes data augmentation to facilitate generalization in inference. The preprocessing used in the training set differs from that used in the test set:

- **Training:** First, random crop with a 4-pixel padding was applied, generating variations in the object's position and emulating scaling effects. A random horizontal inversion with a 50% probability was also incorporated. Subsequently, the image was transformed into a tensor, scaling its values to the range [0, 1], and normalization was applied using the mean and standard deviation specific to the CIFAR-10 dataset to ensure training stability. Finally, random erasing was employed, which hides areas of the image to improve the model's robustness to partial occlusions, forcing it to learn global features of the object.

· **Test:** The images are converted to tensor and normalized using the same mean and standard deviation used during training, thus ensuring a consistent evaluation of the model.

### 3.5 Training with scheduler

To optimize training, three fundamental modifications were introduced. First, one-hot encoding for labels was eliminated, using class indices directly. This is because the CrossEntropyLoss function in PyTorch is optimized to accept indices. Second, a cosine scheduler was implemented. This adjustment reduces the learning rate along a cosine curve, allowing for a smooth descent that facilitates convergence in the later stages of training. Finally, the stochastic gradient descent optimizer was retained, but the weight decay parameter was increased to $5 \times 10^{-4}$ to apply stronger regularization and prevent overfitting.

## 4 Ablation study

It is presented an ablation study that analyzes the impact on the accuracy obtained on the test set using the proposed components. Each of these components has been evaluated in 100 epochs.

The performance of the three proposed architectures is evaluated, along with the use of data augmentation and learning techniques with the Scheluder. Finally, the accuracy of the best component configuration is determined. The results are shown in **Table 1**.

| Model Architecture | Data Aug. | Scheduler | Accuracy (%) |
|---|---|---|---|
| SimpleCNN (Baseline) | - | - | 82.20 |
| ResNet-18 | - | - | 85.45 |
| WideResNet-32 | - | - | 86.45 |
| DenseNet-100 | - | - | 89.40 |
| SimpleCNN | ✓ | - | 87.82 |
| SimpleCNN | - | ✓ | 82.53 |
| **DenseNet-100** | ✓ | ✓ | **93.87** |

Table 1: Ablation study results.

## 5 Conclusions

The results from the ablation study quantify the individual contribution of each component toward improving accuracy on the CIFAR-10 test set. While the baseline model achieved an accuracy of 82.20%, the DenseNet-100 architecture provided the most significant improvement among the three models, reaching 89.40%. Data augmentation techniques and the alternative training strategies also enhanced performance relative to the baseline, yielding accuracies of 87.82% and 82.53%, respectively.

Finally, integrating the superior architecture with both proposed techniques resulted in a peak test accuracy of 93.87%, representing the highest performance achieved in this study.