# Deep Learning Assignment 2- Image Classification using CNN

Rohan Padaya
A1872217
The University of Adelaide
rohanvasant.padaya@student.adelaide.edu.au

## Abstract

*Convolutional Neural Networks (CNNs) have demonstrated exceptional effectiveness in image classification tasks. However, the efficiency of Convolutional Neural Networks (CNNs) in classification problems is significantly impacted by their structure and the hyperparameters used for the construction of the structure. Using the CIFAR-10 dataset, this paper examines how different architectures and hyperparameters influence Convolutional Neural Networks (CNNs) classification capabilities. We compared different models such as the baseline CNN model, the data augmented model, the grid search model, and the ResNet50 model. Our results show that the baseline model yields 70.22% validation accuracy, which is quite decent. After performing data augmentation, the model's validation accuracy improved to 73.23%. Further fine-tuning with a ResNet architecture led to a peak validation accuracy of 81.66%.*

## 1. Introduction

With the advent of Large Neural Networks, the Image classification industry has grown by leaps and bounds particularly using Convolutional Neural Networks (CNN). CNNs are currently regarded as one of the best algorithms for a plethora of Image analysis use cases such as Image recognition, Object detection, and Segmentation [1]. One of the key drivers behind the success of CNNs is the increase in depth of the Network, which has been shown to substantially improve performance [2]. Deep convolutional neural networks have been shown to achieve unprecedented accuracy on the challenging ImageNet dataset, effectively revolutionizing the field of image classification [3]. Furthermore, Residual connections were introduced to address the vanishing gradient problem in deep networks, significantly improving both training efficiency and model performance [4]. Hence, there is countless research available depicting the effectiveness of CNNs in image classification problems.

Despite the advancements in the depth of the network, the efficacy of CNNs also depends on the selection of the model and the hyperparameters used. This paper involves the Image classification of the CIFAR10 dataset using different CNN Architectures and appropriate Hyperparameters for its tuning. The CIFAR10 dataset is a widely popular dataset used for training and testing different CNN models in the contemporary research of Machine learning and Deep Learning Models. We will initially establish a baseline model for the comparison and develop other architectures using different Hyper-parameter settings to improve its performance.

### 1.1. Problem Description

The task at hand is effectively identifying images from the CIFAR-10 dataset using different network architectures and Hyper-parameter tuning. The effectiveness of the models will be measured using the Accuracy of the model as the primary metric.

### 1.2. Dataset Description

The CIFAR-10 Dataset presents a set of unique challenges that make it an ideal candidate for our image classification problem statement. The dataset comprises 60,000 color images, each with dimensions of 32x32 pixels, distributed across 10 distinct classes. These classes include entities such as airplanes, automobiles, birds, cats, deer, dogs, frogs, horses, ships, and trucks, with each class containing a set of 6,000 images. The dataset was initially segmented into five separate training batches, each containing 10,000 images [5]. The dataset was acquired from the following URL: https://www.kaggle.com/datasets/fedesoriano/cifar10-python-in-csv

### 1.3. Method Overview

As mentioned in the Introduction, this study employs Convolutional Neural Network models to classify images of the CIFAR10 dataset. The methodology involves building a baseline CNN model with 8 layers. The data is initially visualized to ensure that the data is balanced in terms

of classes. The data is then pre-processed and rendered to the baseline model. On further analysis, the model is enhanced by finding the best parameters using the Grid search technique and Data augmentation. Additionally, the study builds a ResNet model with L2 Regularization to yield better model accuracy and to avoid over-fitting.

## 2. Method Description

This study uses TensorFlow and Keras libraries to implement and train different Convolutional Neural Network (CNN) models. All the models were run on a specialized external GPU with limited resources.

### 2.1. Data Visualization

One of the most important steps before constructing the models is to visualize the data available. The data was visualized to understand its structure and its distribution. A bar chart (Figure 1) is used to check the balance of the data for each class to ensure that each output category is adequately represented in the data to avoid any biases in the models. A few images from the dataset were displayed to ensure that the data was loaded appropriately with the corresponding labels. Here is the example of the displayed image with its corresponding label in the figure 2

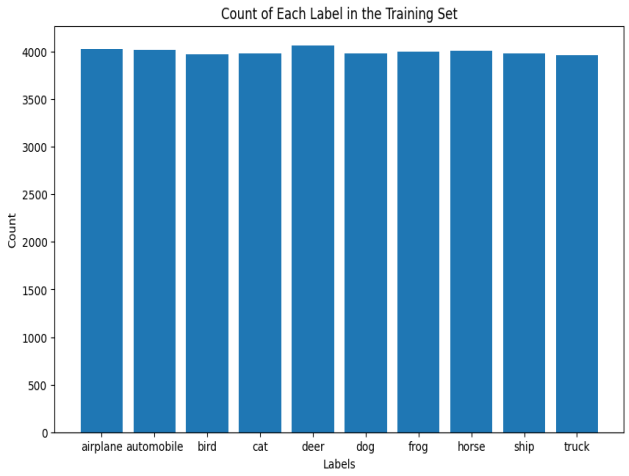


Figure 1. Class Distribution

### 2.2. Data Pre-Processing

The data is split into training, validation, and testing set with an 80/20 (train/val) split. All the images in the sets were normalized to the range[0,1]. This normalization helps in speeding up the convergence of the model. Since we are using sparse categorical cross-entropy as the loss function for our models, there is no need for one hot encoding of the categorical variables in our dataset. However, for models using categorical cross-entropy as the loss function, this step becomes essential.

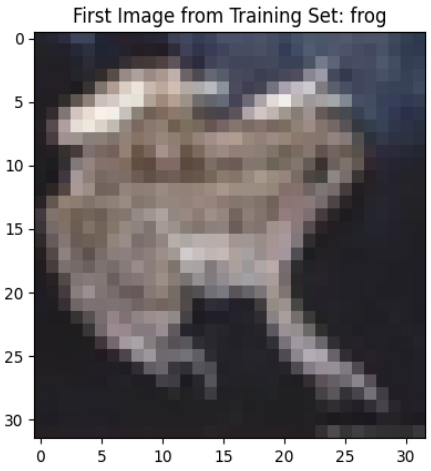


Figure 2. Displaying Image from the Dataset

### 2.3. Baseline Model Architecture

The methodology begins with building a baseline model for a Convolutional Neural Network (CNN) including 8 layers which serves as the foundational model for this study. It comprises three convolutional layers followed by max-pooling layers and two fully connected layers. The convolutional layers use a 3x3 filter (kernel) that slides across the input image to produce a feature map. The convolutional layers also use the ReLU (Rectified Linear Unit) as the activation function. The model ends with the softmax layer with 10 output units representing the ten output classes of the CIFAR10 dataset. Figure 3 shows the detailed structure of our baseline model.

Model: "sequential"

| Layer (type) | Output Shape | Param # |
|---|---|---|
| conv2d_1 (Conv2D) | (None, 30, 30, 32) | 896 |
| max_pooling2d (MaxPooling2D) | (None, 15, 15, 32) | 0 |
| conv2d_2 (Conv2D) | (None, 13, 13, 64) | 18496 |
| max_pooling2d_1 (MaxPooling2D) | (None, 6, 6, 64) | 0 |
| conv2d_3 (Conv2D) | (None, 4, 4, 128) | 73856 |
| flatten (Flatten) | (None, 2048) | 0 |
| dense (Dense) | (None, 128) | 262272 |
| dense_1 (Dense) | (None, 10) | 1290 |

Total params: 356810 (1.36 MB)
Trainable params: 356810 (1.36 MB)
Non-trainable params: 0 (0.00 Byte)

Figure 3. Baseline Model Structure

## 2.4. Data Augmentation

Data Augmentation is a technique of artificially modifying the existing images in the dataset to improve the model's performance by minimizing over-fitting [6]. A survey study on Data Augmentation illustrates the advantages of different types of data augmentation techniques on deep convolutional neural networks [7]. For our model, we have performed random rotation, zooming, and horizontal-flip to increase our dataset. The aim of this step was to reduce over-fitting and to make our baseline model more generalized.

## 2.5. Hyperparameter Optimization using Grid Search

Grid Search is a popular technique in the field of machine learning. This technique involves a full-scale search of the optimal hyperparameters from a grid of parameter values specified by the param_grid parameter for the corresponding model [8]. A study about using grid search for optimization of hyperparameters of machine learning models for prediction of HIV/AIDS test results reveals a positive impact of the grid search technique on the model's prediction accuracy [9]. Hence, To enhance our baseline model further, we conduct a Grid search to find a combination of Hyperparameters that yields the best validation accuracy. The hyperparameters under consideration are the type of optimizer (SGD, Adam), batch size (32, 64), and the number of epochs (10, 20). The Grid Search technique is integrated with the data augmentation approach previously employed, as the augmented data demonstrated notable effectiveness in reducing model over-fitting.

## 2.6. ResNet Model Architecture

ResNet which stands for Residual Network is a special type of deep Convolutional Neural Network that uses a skip connection to solve the Vanishing gradient problem associated with large traditional Convolutional Neural Networks (CNNs) [10]. The model was first introduced in [4] which revolutionized the field of deep learning.

### 2.6.1 Need for ResNet Model

In order to capture complex patterns in the images, there is a common theory in the field of deep learning that the traditional Convolutional network needs to have as many layers as possible. The idea is that the first layer might capture the edges, the second layer might capture the textures, and so on. However, there is a threshold to the number of layers that you can add to a Convolutional Neural Network. As the number of layers increases, the training and test error starts to increase. This problem is called the Vanishing Gradient problem or Exploding Gradient problem wherein the gradient becomes quite small while propagating through all the layers of the network [10]. Figure 4 from [11] illustrates the idea that the higher-layered CNN model (56-layered) results in higher training and testing error rates than a comparatively lower-layered CNN model (20-layered). [11]

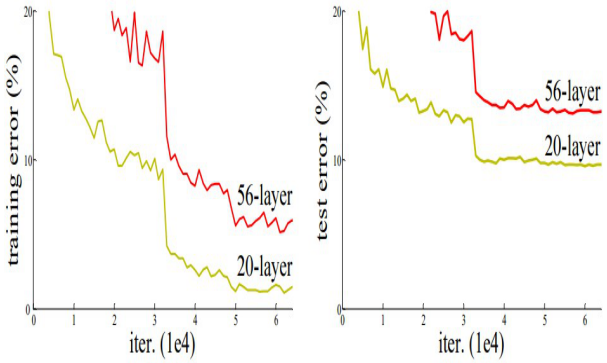


Figure 4. 20-layered vs 56-layered error rates

### 2.6.2 Basic Building Block

The basic building block of a ResNet Architecture is the Residual block. This block includes a skipping connection that bypasses one or more layers allowing the input to be added directly to the output of a later layer [10]. The main purpose of this skipping connection is to mitigate the Vanishing gradient problem or the Exploding gradient problem [11]. Figure 5 from [9] shows the structure of the basic building block and the skipping connection of the ResNet architecture.

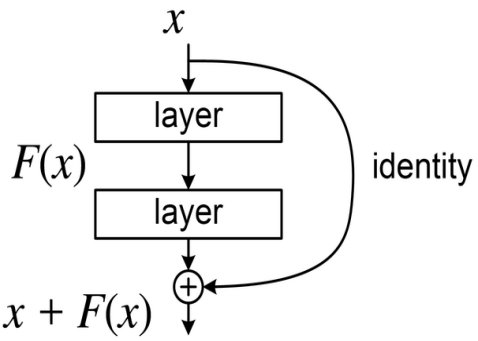


Figure 5. Basic Building Block of ResNet

### 2.6.3 Our ResNet Implementation

The architecture of ResNet-50 is structured into five distinct phases, with each phase comprising a different number of residual units. The first stage, known as conv2, includes 3 residual blocks, while the second stage, conv3, has 4. The

third stage, conv4, consists of 6 blocks, and the fourth stage, conv5, has 3. Each of these blocks incorporates three convolutional layers, and the final layer employs a softmax activation function. This results in a cumulative total of 50 convolutional layers throughout the network.

## 2.7. Experimental Analysis

### 2.7.1 Primary Objective

The primary objective of this study is to analyze different Convolutional Neural Network Architectures and their corresponding hyper-parameters tuning for the image classification of the CIFAR-10 dataset. The models mentioned in the methodology were extensively analyzed under various settings to identify the optimal solution for our use case. The analysis involves Data Visualization, Data Pre-Processing, Baseline Modeling, Data Augmentation over the baseline model, Grid search over the baseline model to identify the best parameters, ResNet50 modeling, and adding regularization to ResNet50 architecture to further improve its performance. In the results section, we will explain the steps taken and the results achieved through those steps in chronological order.

### 2.7.2 Results

**Data Visualization:** Initial visualization of the dataset using a bar graph (Figure 2) showed that the CIFAR-10 dataset is balanced across all ten classes, each containing **4,000 images in the training data**. Several other visualizations were carried which will be covered in the future sections.

**Data Pre-Processing:** In the pre-processing stage, the training data was split into training and validation sets using the train_test_split package from the sklearn standard library with an 80/20 (train/val) split. After the split, the training data had (40000, 32, 32, 3) shape which indicates 40000 images, 32 height, 32 width, and 3 color channels. Both the validation and test sets had (10000, 32, 32, 3) shape which indicates 10000 images, 32 height, 32 width, and 3 color channels. The pixel values of each image were normalized and scaled for data modeling.

**Baseline Modeling:** In the baseline model, as we can see in the figures 6 and 7, the training accuracy showed a consistent increase, and the training loss showed a consistent decrease. However, the validation accuracy seems to increase for a while then starts to decrease again in the end. Similarly, the validation loss decreases for most epochs but then starts to increase again. This is indicative of the over-fitting of the model. Also. the significant difference in the training accuracy (84.11%) and validation accuracy (70.04%) suggests that the model is clearly over-fitting. To improve this, we decided to use the Data Augmentation technique which is known to mitigate the over-fitting problem.

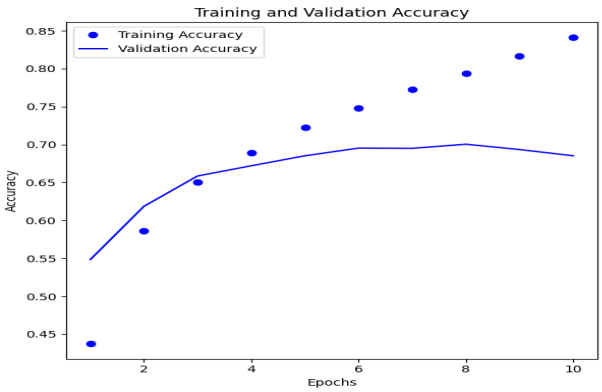


Figure 6. Accuracy curve of baseline model

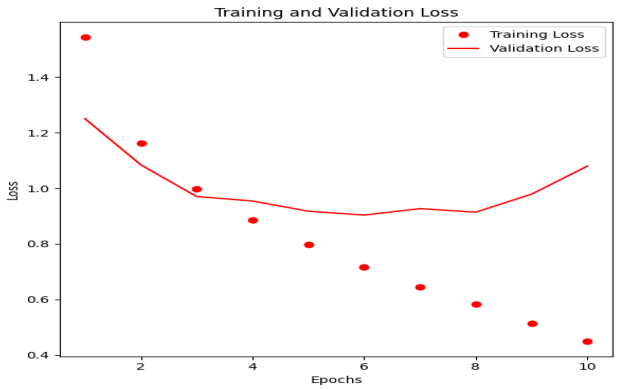


Figure 7. Loss curve of baseline model

**Data Augmentation:** In the Data Augmentation phase, We applied rotation, horizontal flipping, and zooming transformations to augment the data and create more unseen images for the model to perform better generalization and avoid over-fitting. We plotted the same Accuracy vs Loss curves for training and validation sets of data augmented model. The curves reflect considerable improvement in the model's performance. In fact, the validation accuracy is higher (73.45%) as compared to the training accuracy (71.70%). This suggests that the model is not over-fitting and it is generalizing well to the unseen data. Figures 8 and figures 9 are reflecting the nature of the curves. The training curves for accuracy and loss show a typical trend, with accuracy increasing and loss decreasing as training progresses, implying effective learning from the dataset. In contrast, the validation curves for both accuracy and loss display a fluctuating pattern. This could point to the model's sensitivity to particular validation batches. However, despite these fluctuations, the validation accuracy ultimately reaches at a high

point, suggesting that the model is capable of generalizing effectively to unseen data.

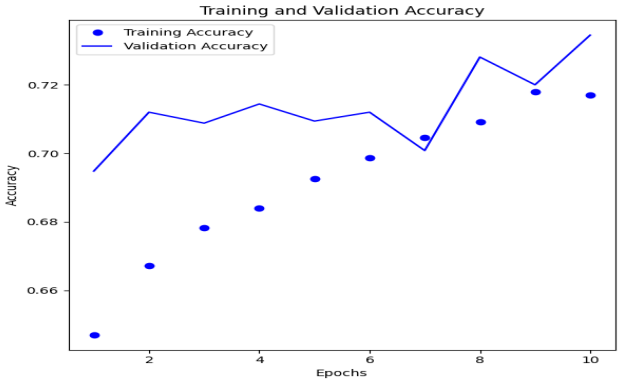


Figure 8. Accuracy curve of Data Augmented model


Figure 9. Loss curve of Data Augmented model

**Grid search Modeling:** Using the Grid search technique, we identified the best hyperparameters for our baseline model to further enhance its accuracy. Since we have fixed the over-fitting issue of the model, it would be ideal to improve the accuracy further by experimenting with different parameters. We used our previous data-augmented model and applied the Grid search technique on the model using loops. The hyperparameters under consideration are the type of optimizer, the number of epochs, and the batch size. The best parameters identified are Adam optimizer, 20 epochs, and a batch size of 32. The best validation accuracy achieved under the Grid search technique is 72.18%. These hyperparameters will be used in further complex models.

**ResNet Modeling:** As mentioned previously, Complex models like ResNet have the ability to improve the Convolutional Neural Network models in a significant way by mitigating the Vanishing/Exploding Gradient problem. There are numerous types of ResNet models depending on the number of layers in the model. We trained a ResNet model with 50 layers (ResNet50) which achieved a peak validation accuracy of 75.08%. This accuracy that we achieved is not too different from the accuracies achieved by data augmented model (73.45%) and the Grid search model (72.18%). However, the model seems to generalize well to the unseen data as there is not much difference between the training accuracy (79.11%) and the validation accuracy (75.08%). There is still scope for further experimentation on the ResNet model to improve its accuracy.

**ResNet Model with L2 Regularization:** As part of our analysis, we decided to experiment further with the ResNet50 model by adding an L2 regularization component in the model for improvement in the accuracy of the model. The ResNet model with the regularization component showed a significant improvement in both the training accuracy and validation accuracy of the model. The highest training accuracy achieved was 99.46% and the highest validation accuracy achieved was 80.66% which is way higher than all the other models. We evaluated the model on the testing set as well to confirm whether the model does well for unseen data. The test accuracy of the model turned out to be 79.73% which suggests that the model performs well for the test data.

Table 1. Training Accuracies of Different Models

| Model | Training Accuracy (%) |
|---|---|
| Baseline | 84.11 |
| Data Augmented | 73.21 |
| Grid Search Best | 72.98 |
| ResNet | 79.11 |
| ResNet with L2 Regularization | 99.46 |

Table 2. Validation Accuracies of Different Models

| Model | Validation Accuracy (%) |
|---|---|
| Baseline | 70.04 |
| Data Augmented | 73.23 |
| Grid Search Best | 72.18 |
| ResNet | 72.85 |
| ResNet with L2 Regularization | 80.66 |

## 2.8. Code

The code is implemented using Python framework executed on specialized GPU in a Google Collab environment and relies on the TensorFlow and Keras libraries for building the Convolutional Neural Network models. The code is uploaded on github public repository. Here's the link to the same: https://github.com/rohanpadaya/Deep-Learning.git

## 2.9. Conclusion

This study investigated the influence of different Network Architectures and Hyperparameter settings on the Convolutional Neural Network (CNN) Models. The study involved Image classification of the CIFAR-10 dataset employing multiple CNN models to check their effectiveness in this use case.

The findings of the study suggest that the data augmentation model proved effective in mitigating over-fitting, leading to a more generalized model. On performing Grid-search, the hyper-parameter tuning gave insights into the optimal parameters for the model. These hyperparameters were used to train a more complex ResNet model with and without the L2 regularization. Interestingly, the ResNet50 Model with L2 regularization demonstrated superior performance among all the models achieving a test accuracy of 79.73%.

Despite the promising results, there is further scope in experimenting with more complex models like Visual Graphics Group (VGG), EfficientNet, and other complex architectures. Due to limited time and resources, this study does not further explore these architectures.

In conclusion, this research offers an in-depth analysis of Convolutional Neural Networks for the task of image classification, offering valuable insights into model structures, the significance of hyper-parameter optimization, as well as the impact of data augmentation and regularization techniques. These findings can serve as a reference for future studies in this rapidly advancing domain of deep learning.

## References


[1] M. Tripathi, "Image processing using cnn — beginner's guide to image processing," Analytics Vidhya, 06 2021. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/06/image-processing-using-cnn-a-beginners-guide/ 1

[2] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, and V. Vanhoucke, "Going deeper with convolutions," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9, 2015. [Online]. Available: https://www.cs.unc.edu/~wliu/papers/GoogLeNet.pdf 1

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, pp. 84–90, 05 2012. [Online]. Available: https://dl.acm.org/doi/10.1145/3065386 1

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, pp. 1904–1916, 09 2015. 1, 3

[5] A. A. Awan, "A complete guide to data augmentation," www.datacamp.com, 11 2022. [Online]. Available: https://www.datacamp.com/tutorial/complete-guide-data-augmentation 1

[6] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, 07 2019. [Online]. Available: https://journalofbigdata.springeropen.com/articles/10.1186/s40537-019-0197-0 3

[7] "3.2. tuning the hyper-parameters of an estimator — scikit-learn 0.22 documentation," Scikit-learn.org, 2012. [Online]. Available: https://scikit-learn.org/stable/modules/grid_search.html 3

[8] D. M. Belete and M. D. Huchaiah, "Grid search in hyperparameter optimization of machine learning models for prediction of hiv/aids test results," *International Journal of Computers and Applications*, pp. 1–12, 09 2021. 3

[9] W. Contributors, "Residual neural network," Wikipedia, 08 2019. [Online]. Available: https://en.wikipedia.org/wiki/Residual_neural_network 3

[10] T. SHARMA, "Detailed explanation of residual network(resnet50) cnn model." Medium, 03 2023. [Online]. Available: https://medium.com/@sharma.tanish096/detailed-explanation-of-residual-network-resnet50-cnn-model-106e0ab9fa9e 3

[11] "Residual networks (resnet) - deep learning," GeeksforGeeks, 06 2020. [Online]. Available: https://www.geeksforgeeks.org/residual-networks-resnet-deep-learning/ 3