# Efficiencies of Hyper Parameters in CNN during Image Training

Hongjo Yoon

March 2024

**Abstract**

This project investigates the effectiveness of convolutional neural networks (CNN) during image classification tasks, utilizing the CIFAR-10 dataset. We explore two popular convolutional neural network architectures, ResNet and GoogLeNet in order to compare the different performances on the same dataset in order to determine which one should be used on image classification. In order to determine what architecture has the best performance, firstly we must determine the optimal parameters such as number of layers (such as 18, 34, or 50 for ResNet and having the default amount of layers for GoogLeNet), different pooling options (max pooling vs average pooling), different optimization methods such as Adam vs stochastic gradient descent, and different activation functions of ReLU and Sigmoid. This project aims to help the world of image training by finding out the different kinds of options that someone may have when first beginning, helping find a starting point for similar projects.

## 1 Introduction

In today's world of machine learning there are many ways that one can train a image dataset using various different techniques, however certain techniques require certain parameters and options in order to be optimal saving both effort and time. What we want to see in our results for this project is the different accuracies that can be found when trying to train a convolutional neural network on an image dataset using different pooling options, optimization methods, etc. in order to find the most optimal combined set of options, in this case what set of options work best when using ResNet and GoogLeNet.

## 2 Methodology

### 2.1 Procedures

In order to be able to determine which set of parameters are optimal to train images on we manually ran the code for ResNet and GoogLeNet changing the parameters as we went on such as the amount of layers, in order to see which would perform the best with the best accuracy. We downloaded the CIFAR-10 dataset and transformed the image values by converting the images into tensors which are then normalized for ease of use. Next all of the parameters are chosen, choosing the amount of layers, the pooling type, the optimization type, and the activation type. After that the image data is then trained on either the ResNet or GoogLeNet architecture which is then tested to get the test accuracy for the entire dataset and for each individual class (which contain the type of image which includes: plane, car, bird, cat, deer, dog, frog, horse, ship, truck). That trained data is also evaluated by showing the images then predicting what class the image belongs to.

## 2.2 ResNet

ResNet is a cnn architecture that adds residual connections which allows data to skip layers and output instead of going through the layers. ResNet utilizes residual blocks to do this operation and adding more layers ends up utilizing more residual blocks allowing for increased efficiency at the cost of more computing power. Additionally adding more layers has the potential to face overfitting the data, which is something that we wish to avoid. The layers that we chose to test were 18, 34, and 50. While we could have used higher amounts of layers such as 152 we thought that this was too computationally taxing for the resources that we had at hand and that it would take too long to compute.

### 2.2.1 Number of Layers

Choosing the number of layers can directly affect the training performance of the algorithm as more and more layers can allow for higher accuracy on larger amounts of data, however that comes at a price as having too many layers can lead to overfitting and have a larger strain on resources, creating limitations on weaker computers if these trials were to be used by another person.

## 2.3 Optimization Methods

The different optimization methods that we chose were adaptive moment estimation (adam) and stochastic gradient descent. Adam has the ability to adapt the learning rate for every parameter individually meaning that it is able to converge in less time compared to sgd, meaning that it saves time and resources when using it. Adam is able to perform correction on the bias of the architecture meaning that it is able to come to a more accurate conclusion. SGD is able to utilize the entire dataset in order to use gradient descent allowing for ease of use for larger datasets than adam but comes at the cost of higher computational power and longer processing times.

### 2.4 Pooling Functions

In order to reduce the dimensions of the dataset we decided to use two different pooling methods, max pooling and average pooling. Max pooling looks at the input feature and selects the maximum value of that windowed feature and is then able to extract only the maximum values of the dataset meaning that the most important features are prominent. Average pooling, similar to max pooling, looks at the data through a window of a square size of data, and then computes the average in order to reduce the size of the sample. This allows the cnn to be able to reduce the negative effects of noise better than max pooling, however max pooling is able to have more detailed results.

### 2.5 Activation Functions

The activation functions that we used for our cnns were ReLU and Sigmoid. ReLU is able to reduce the computational requirements to run the cnn allowing for faster training times. The way that ReLU works is that it takes the data of the images and returns a 1 if the data is positive after putting it in the ReLU formula which is defined as f(x) = max(0, x) and returns 0 otherwise. The other activation function, Sigmoid as defined by the function $f(x) = \frac{1}{1+e^{-x}}$ which results in a value between 0 and 1. Sigmoid allows the cnn to be able to have a result of a probability that the data fits the cluster, which helps the cnn be able to predict.

### 2.6 GoogLeNet

GoogLeNet is the other architecture that we used to train the cnn on CIFAR-10 and was designed by Google, hence the name GoogLeNet. GoogLeNet utilizes inception modules which perform multiple convolutions all with varying sizes on the same layer which allows better efficiency in training. This is especially useful in an image dataset due to the possibility of differing image sizes. Adding more layers would allow for better representation in a more complex dataset, however adding too many layers could introduce the problem of overfitting which would end up hurting the result. In our case however we decided to just add two more layers when comparing to no additional layers to compare performance. As for the other hyper-parameters, such as the activation functions, pooling functions, and optimization method, those methods should have similar effect to GoogLeNet as they did to ResNet.

## 3 Dataset

The dataset that we used was the CIFAR-10 dataset. The CIFAR-10 dataset is a popular benchmark dataset that contains 60000 images of size 32x32 with 50000 images meant for training and 10000 images meant for testing. This is a public dataset created by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. This dataset contains 10 classes which are the type of object that the image is of. The classes are airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck.

# 4 Trials

For our trials, as mentioned previously, all of the interchangeable options are changed by hand and simply trained and tested in order to gain the test accuracy for each combined options. We made sure to test every combination of pooling method, activation function, etc. The results were numerical and included testing accuracies for not just the entire dataset, but for each class as well.

## 4.1 ResNet 18 Layers

Throughout our trials that we conducted for ResNet with 18 layers we found that utilizing Adam, maximum pooling, and using ReLU gave us the best overall accuracy when testing on 10,000 images. We got a 56% accuracy rate overall but for each class some other combination of parameters ended up having higher accuracies. For example while adam, maximum pooling, and ReLU has the highest overall accuracy but for the dog class the combination of Adam, Maximum pooling, and sigmoid had a higher accuracy rate by 3%. Below is a bar graph showing the accuracy rates for all the different configurations.

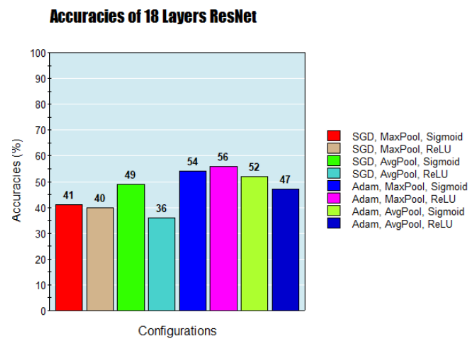


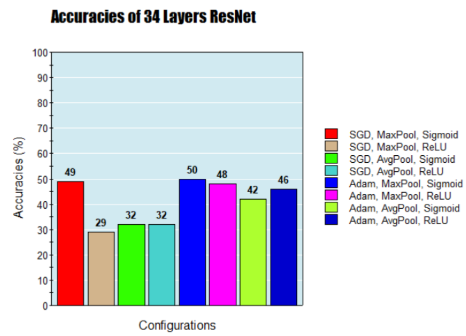

Figure 1:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Plane | 62 | 21 | 42 | 48 | 69 | 63 | 58 | 50 |
| Car | 65 | 48 | 51 | 58 | 70 | 59 | 67 | 64 |
| Bird | 18 | 7 | 28 | 26 | 38 | 42 | 34 | 24 |
| Cat | 20 | 35 | 49 | 21 | 35 | 39 | 25 | 42 |
| Deer | 46 | 53 | 63 | 18 | 47 | 46 | 44 | 44 |
| Dog | 45 | 38 | 21 | 27 | 40 | 37 | 51 | 28 |
| Frog | 26 | 33 | 55 | 49 | 61 | 73 | 65 | 52 |
| Horse | 54 | 47 | 50 | 42 | 63 | 56 | 55 | 49 |
| Ship | 31 | 77 | 60 | 38 | 63 | 72 | 63 | 62 |
| Truck | 45 | 38 | 75 | 39 | 58 | 69 | 59 | 55 |

Table 1: Test Accuracy for Each Class in % (18 Layers) (SGD, MaxPool, Sigmoid), (SGD, MaxPool, ReLU), (SGD, AvgPool, Sigmoid), (SGD, AvgPool, ReLU), (Adam, MaxPool, Sigmoid), (Adam, MaxPool, ReLU), (Adam, AvgPool, Sigmoid), (Adam, AvgPool, ReLU)

## 4.2 ResNet 34 Layers

For 34 layers in ResNet we found that using Adam, Maximum Pooling, and sigmoid, had the highest accuracy of 50% however we found that it was only highest by a few percent, as SGD, maximum pooling, and sigmoid, had a accuracy rate of 49% and Adam, maximum pooling, ReLU had a rate of 48% Below is a bar graph showing the accuracy rates for all the different configurations.




| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Plane | 62 | 48 | 43 | 39 | 67 | 59 | 50 | 63 |
| Car | 65 | 43 | 24 | 46 | 68 | 68 | 52 | 67 |
| Bird | 36 | 19 | 13 | 7 | 38 | 34 | 22 | 25 |
| Cat | 21 | 9 | 11 | 21 | 34 | 39 | 38 | 22 |
| Deer | 28 | 26 | 13 | 42 | 35 | 40 | 27 | 31 |
| Dog | 45 | 35 | 39 | 19 | 34 | 26 | 17 | 37 |
| Frog | 56 | 20 | 59 | 30 | 64 | 53 | 52 | 60 |
| Horse | 58 | 41 | 27 | 35 | 62 | 56 | 49 | 60 |
| Ship | 63 | 38 | 34 | 28 | 54 | 64 | 56 | 49 |
| Truck | 55 | 16 | 49 | 54 | 50 | 42 | 56 | 51 |

Table 2: Test Accuracy for Each Class in % (34 Layers) (SGD, MaxPool, Sigmoid), (SGD, MaxPool, ReLU), (SGD, AvgPool, Sigmoid), (SGD, AvgPool, ReLU), (Adam, MaxPool, Sigmoid), (Adam, MaxPool, ReLU), (Adam, AvgPool, Sigmoid), (Adam, AvgPool, ReLU)

### 4.3 ResNet 50 Layers

For 50 layers in ResNet we found that the optimal hyper-parameters were Adam, Maximum Pooling, and ReLU with a test accuracy of 58% on 10,000 images. The next closest was at 56% using Adam, Maximum Pooling and Sigmoid.

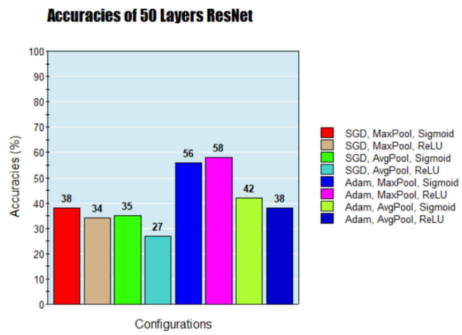



Figure 2:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Plane | 38 | 44 | 41 | 44 | 58 | 60 | 43 | 45 |
| Car | 57 | 44 | 57 | 40 | 70 | 71 | 49 | 68 |
| Bird | 14 | 10 | 26 | 18 | 35 | 43 | 31 | 10 |
| Cat | 15 | 3 | 9 | 2 | 30 | 50 | 13 | 10 |
| Deer | 27 | 41 | 11 | 27 | 44 | 52 | 29 | 22 |
| Dog | 34 | 37 | 40 | 17 | 51 | 32 | 37 | 39 |
| Frog | 61 | 42 | 49 | 32 | 67 | 67 | 59 | 55 |
| Horse | 53 | 30 | 35 | 34 | 63 | 66 | 48 | 47 |
| Ship | 47 | 40 | 41 | 33 | 70 | 71 | 56 | 57 |
| Truck | 38 | 47 | 44 | 24 | 71 | 73 | 56 | 23 |

Table 3: Test Accuracy for Each Class in % (50 Layers) (SGD, MaxPool, Sigmoid), (SGD, MaxPool, ReLU), (SGD, AvgPool, Sigmoid), (SGD, AvgPool, ReLU), (Adam, MaxPool, Sigmoid), (Adam, MaxPool, ReLU), (Adam, AvgPool, Sigmoid), (Adam, AvgPool, ReLU)

## 4.4 GoogLeNet

After trying out all of the different hyper-parameter combinations of GoogLeNet, we found that using SGD, Maximum Pooling, and ReLU, ended up with the highest testing accuracy with an accuracy of 54%. Another close one happened to be using SGD, Maximum Pooling, and Sigmoid, with an accuracy of 51%. The next closest one was 42% but after that there seemed to be a significant drop to 37% and lower making them not as viable options as the first two that were mentioned. Below is a bar graph to visualize the different hyper-parameter results. Something interesting to note is that GoogLeNet seemed to perform poorly when testing bird and cat images and especially with the parameters (Adam, Maximum Pooling, Sigmoid), (Adam, Average Pooling, Sigmoid), and (Adam, Average Pooling, ReLU)

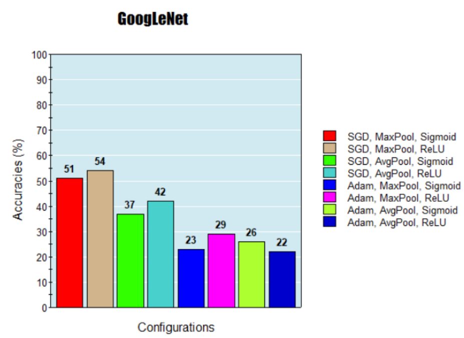



Figure 3:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Plane | 67 | 64 | 34 | 51 | 37 | 24 | 38 | 26 |
| Car | 39 | 66 | 48 | 52 | 36 | 25 | 42 | 19 |
| Bird | 31 | 35 | 6 | 30 | 13 | 9 | 3 | 6 |
| Cat | 22 | 33 | 13 | 22 | 1 | 5 | 8 | 28 |
| Deer | 45 | 48 | 44 | 26 | 7 | 27 | 13 | 2 |
| Dog | 49 | 43 | 38 | 26 | 40 | 55 | 23 | 6 |
| Frog | 63 | 58 | 46 | 62 | 30 | 29 | 44 | 41 |
| Horse | 58 | 62 | 38 | 45 | 7 | 30 | 26 | 26 |
| Ship | 60 | 64 | 58 | 67 | 31 | 46 | 34 | 53 |
| Truck | 77 | 65 | 39 | 42 | 34 | 40 | 30 | 12 |

Table 4: Test Accuracy for Each Class (GoogLeNet) (SGD, MaxPool, Sigmoid), (SGD, MaxPool, ReLU), (SGD, AvgPool, Sigmoid), (SGD, AvgPool, ReLU), (Adam, MaxPool, Sigmoid), (Adam, MaxPool, ReLU), (Adam, AvgPool, Sigmoid), (Adam, AvgPool, ReLU)

# 5 Conclusion

After looking at all of the test accuracies for each of the trials we can conclude that out of the different options, having 50 layers, using Adam, using Maximum Pool, and ReLU ended up being the most optimal set of parameters and options for ResNet when training images. As for GoogLeNet, we found that using SGD, Maximum Pooling, and ReLU ended up being optimal with a accuracy rate of 54%. We hope that our project ends up being useful for the world of machine learning and the world of image training so that people in the future who want to train images using ResNet or GoogLeNet are able to save time and computational power and possible computing costs as well, due to the fact that they would be able to look at our data and consider the options with the highest accuracies, so that they know where to start.

# 6 Acknowledgments

We thank professor Zhuowen for the machine learning education and the skeleton code that he provided for us in one of our homeworks, which we utilized in training and grabbing the data from Cifar-10.

# 7 References

Alex Krizhevsky. Learning Multiple Layers of Features from Tiny Images. April 8, 2009. March 23, 2024. Professor Zhuowen. Assignment 4.