Introduction to Machine Learning Final Report

Kyle Keeton[1] and Syed Rizvi[2]

[1]ID: KHK210001 , email: KHK210001@utdallas.edu , Department: Computer Science , contributions: report formatting, report writing, LeNet architecture, LeNet modifications, dataset exploratory analysis
[2]ID: SAR180000 , email: SAR180000@utdallas.edu , Department: Computer Science , contributions: AlexNet, hyperparameter tuning, model training, metrics, graphs, dataset preprocessing, report proofreading

Contents

I. Abstract

While only capable of processing grayscale images, LeNet does not require a high amount of computational resources compared to colored image classifiers. As a result, it is beneficial for use in educational settings where computational resources are limited and the focus is on understanding concepts of CNN's rather than model accuracy.

This paper outlines our experiments and results in modifying the LeNet architecture to process color images. We compare the modified LeNet, called LeNet_RGB, to its original parent architecture to test for any degradation of performance. We also compare LeNet_RGB to AlexNet in performance over the CIFAR-10 and CIFAR-100 datasets to see its performance against a heavier, proven algorithm.

II. Introduction

In this set of experiments, we decided to explore how we can modify LeNet to handle RGB images. As students without access to powerful machines, we have found ourselves to be limited in the neural networks we can create because we do not have the computational resources to train heavy architectures. While the overall accuracy of an algorithm is important to those learning about machine learning, it is irrelevant when they are unable to run the algorithm in question. As it can be beneficial for students to explore algorithms with multiple input layers, we decided to explore modifying the LeNet algorithm for color images. We chose LeNet due to its low computational cost.

When deciding what algorithm to test our LeNet_RGB against, we tested multiple algorithms to find the most computationally expensive convolutional neural network we could run on our hardware within a reasonable time period. After benchmarking AlexNet, VGGNet and ResNet, we settled on AlexNet.

III. Related Work

We explored LeNet [2], VGG [3], AlexNet [1], and ResNet [4] on different image classification datasets.

While LeNet [2] works well in classifying the MNIST dataset, it is unable to handle RGB datasets at all without some extensive preprocessing.

VGG [3] and ResNet [4] were prohibitively computationally expensive to reasonably run and test on our hardware.

AlexNet [1] can successfully handle RGB image datasets at reasonable accuracy. However, it has been surpassed by more modern neural networks.

Demonstrating to students a method of performing small modifications on an existing neural network to create new functionality is vital to help them better understand the concept and expand on future research. Additionally, creating a network that can more easily be run allows students to better explore the field of Machine Learning to eventually contribute more to research in the field.

IV. Proposed Approach

In this set of experiments, we used the following neural nets with the specified parameters:
LeNet [6]:{
{ cnn layers}: Sequential{

(0): Conv2d(1, 6, kernel_size = 5, stride=1, padding = 0)
(1): BatchNorm2d(6)
(2): ReLU()
(3): MaxPool2d(kernel_size=2, stride=2)
(4): Conv2d(6, 16, kernel_size = 5, stride=1, padding = 0)
(5): BatchNorm2d(16)
(6): ReLU()
(7): MaxPool2d(kernel_size=2, stride=2)
}
(Linear Layers): Sequential{
(0): Linear(in_features=400, out_feature=120)
(1): ReLU(inplace=True)
(2): Linear(in_features=120, out_features=84)
(3): ReLU()
(4): Linear(in_features=84, out_features=num_classes)
}
}

LeNet_RGB:{
{cnn layers}: Sequential{
(0): Conv2d(3, 13, kernel_size = 5, stride=1, padding = 0)
(1): BatchNorm2d(13)
(2): ReLU()
(3): MaxPool2d(kernel_size=2, stride=2)
(4): Conv2d(13, 60, kernel_size = 5, stride=1, padding = 0),
(5): BatchNorm2d(60),
(6): ReLU(),
(7): MaxPool2d(kernel_size=2, stride=2)
}
(Linear Layers): Sequential{
(0): Linear(in_features=1500, out_feature=360)
(1): ReLU(inplace=True)
(2): Linear(in_features=360, out_features=256)
(3): ReLU()
(4): Linear(in_features=256, out_features=num_classes)
}
}

AlexNet [5]: {
(features): Sequential{
(0): Conv2d(3, 64, kernel_size=(11, 11), stride=(4, 4), padding=(2, 2))
(1): ReLU(inplace=True)
(2): MaxPool2d(kernel_size=3, stride=2, padding=0, dilation=1, ceil_mode=False)
(3): Conv2d(64, 192, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2))
(4): ReLU(inplace=True)
(5): MaxPool2d(kernel_size=3, stride=2, padding=0, dilation=1, ceil_mode=False)
(6): Conv2d(192, 384, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
(7): ReLU(inplace=True)
(8): Conv2d(384, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
(9): ReLU(inplace=True)
(10): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
(11): ReLU(inplace=True)
(12): MaxPool2d(kernel_size=3, stride=2, padding=0, dilation=1, ceil_mode=False)
}
(avgpool): AdaptiveAvgPool2d(output_size=(6, 6))
(classifier): Sequential{
(0): Dropout(p=0.5, inplace=False)
(1): Linear(in_features=9216, out_features=4096, bias=True)
(2): ReLU(inplace=True)
(3): Dropout(p=0.5, inplace=False)
(4): Linear(in_features=4096, out_features=4096, bias=True)
(5): ReLU(inplace=True)
(6): Linear(in_features=4096, out_features=num_classes, bias=True)
}
}

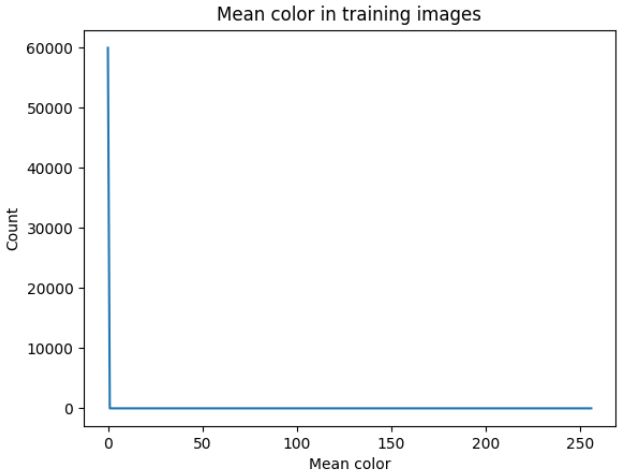
LeNet_RGB was reached by initially multiplying all parameters in LeNet by a factor of 3. Next, we fine tuned various parameters for better results with the CIFAR-10 dataset. Preliminary testing showed this to outperform only changing the input of the first convolutional layer.

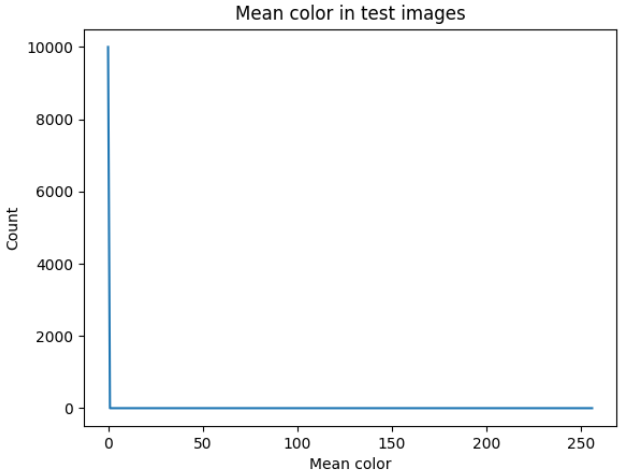
We evaluate the performance of both Adam and SGD optimization algorithms for each model with learning rate 0.01, 0.005, 0.001, 0.0005, and 0.0001. Batch size was always set to 32. After assessing various combinations, we identify the most promising ones. For these selected combinations, we trained to 20 epochs. We then selected the best results between the 2 optimizers.

V. Experiments

A. Exploratory Data Analysis

1) MNIST

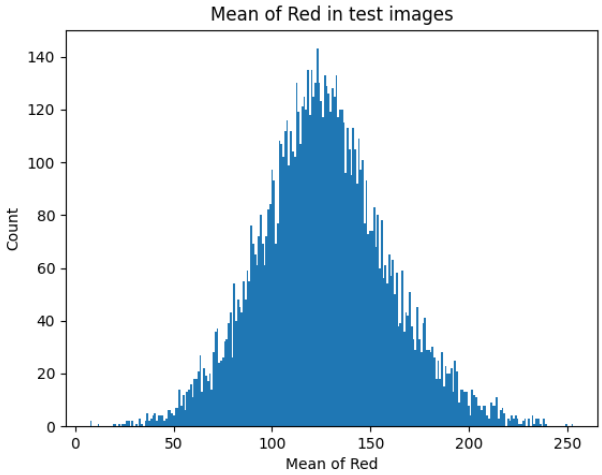


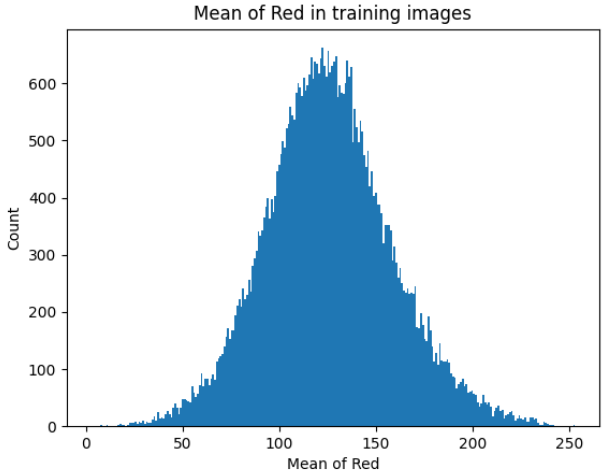


Distribution of average color in all MNIST images

In our exploratory data analysis for the MNIST dataset, we found that the mean color was extremely close to 0. As a result, it can be concluded that the data is almost entirely black.

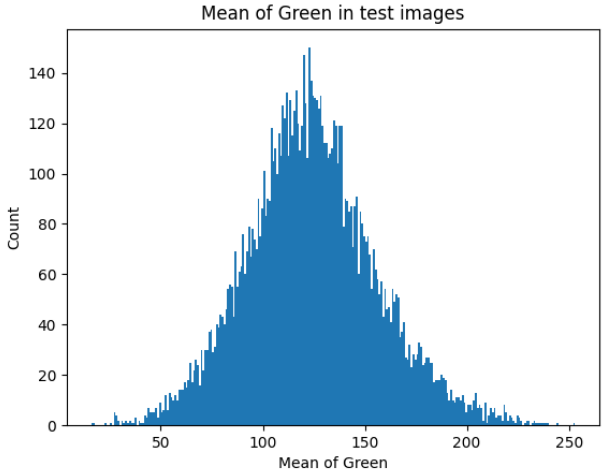
2) CIFAR10

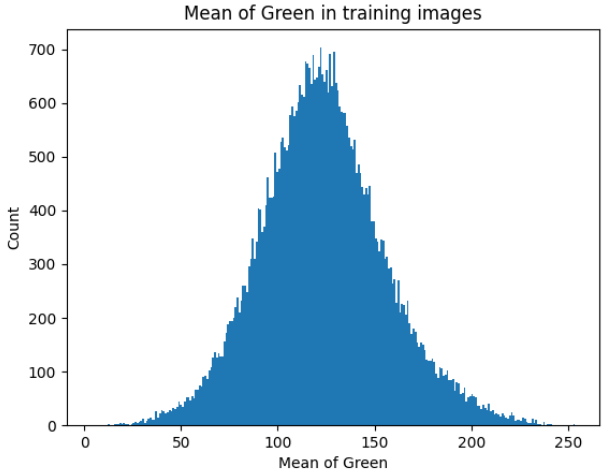




Distribution of average amount of red in all CIFAR10 images

In our exploratory analysis for the average amount of red in each image of the CIFAR10 dataset, we found that the data was approximately normally distributed around the mean of 125. Because there is only one peak, it can be concluded that all categories likely contain a similar amount of red.

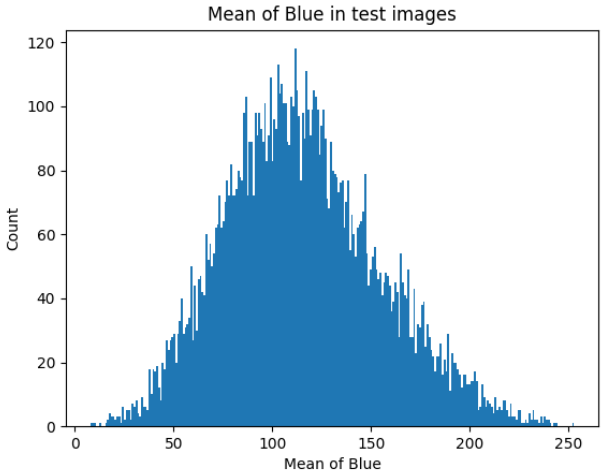


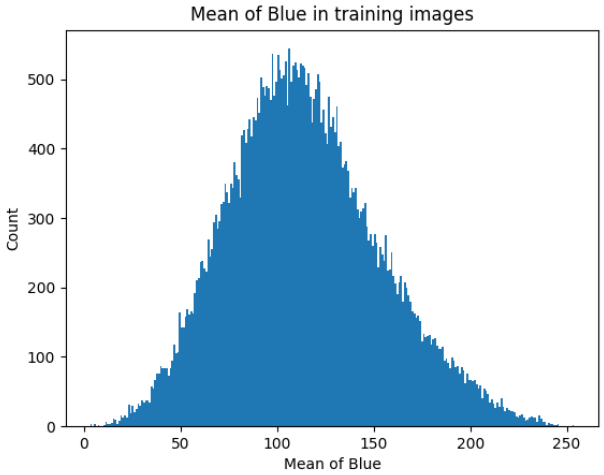


Distribution of average amount of green in all CIFAR10 images

In our exploratory analysis for the average amount of green

in each image of the CIFAR10 dataset, we found that the data was approximately normally distributed around the mean of 125. While the overall distribution of both the test and training datasets were the same, there were more points where the data was sparse for the testing dataset than the training dataset.

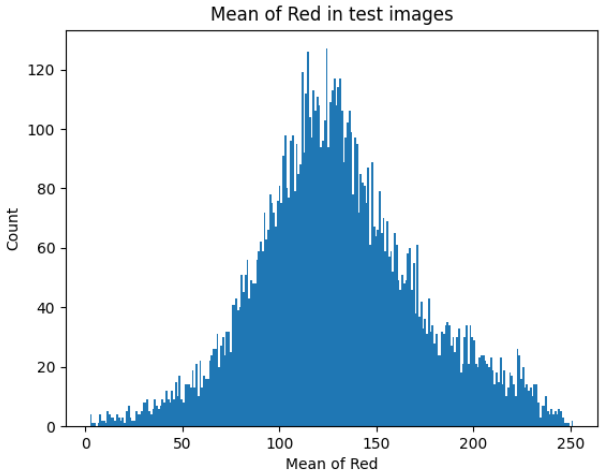


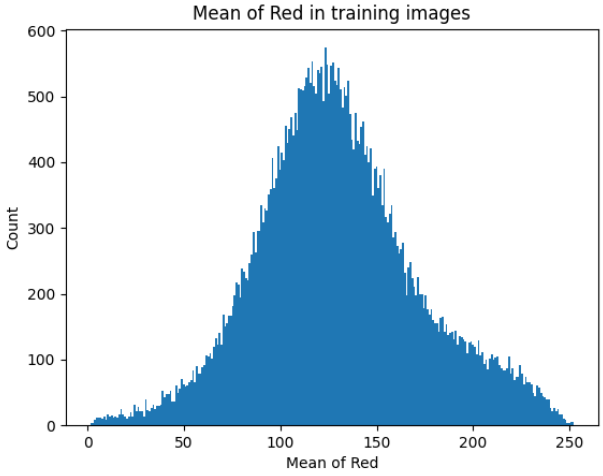


Distribution of average amount of blue in all CIFAR10 images

In our exploratory analysis for the average amount of blue in each image of the CIFAR10 dataset, we found that the data was skewed left with most data clustered around 100. As a result, it can be concluded that most images have less blue than the other main colors.

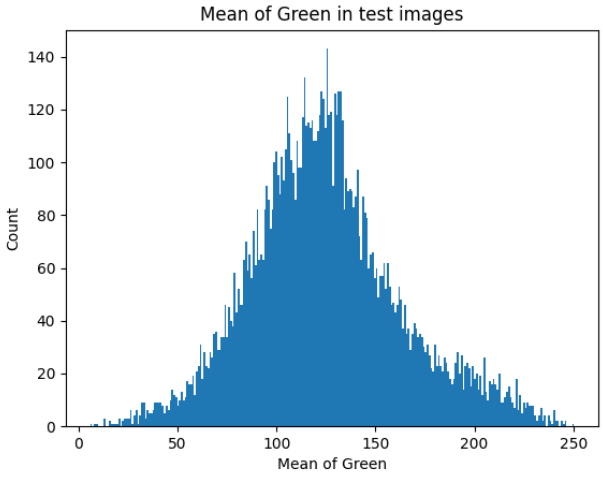
3) CIFAR100

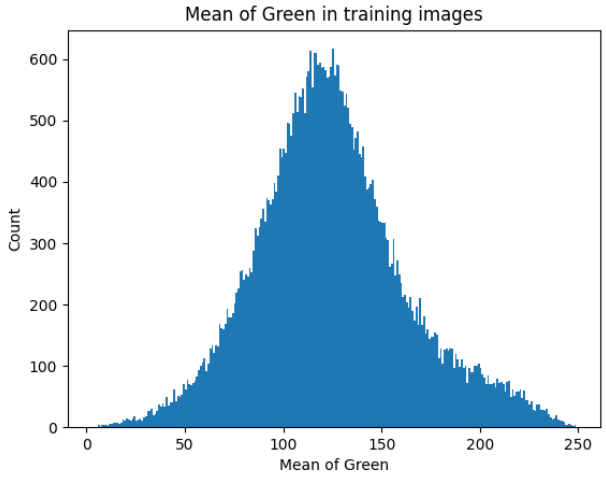




Distribution of average amount of red in all CIFAR100 images

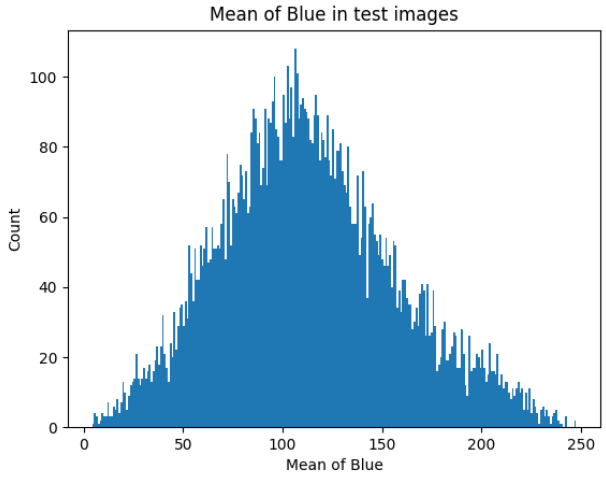
In our exploratory analysis for the average amount of red in each image of the CIFAR100 dataset, we found that the data was approximately normally distributed around the mean of 125. Because there is only one peak, it can be concluded that all categories likely contain a similar amount of red.

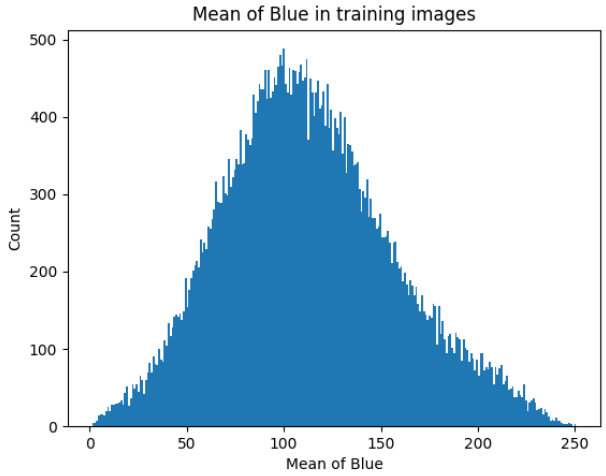




Distribution of average amount of green in all CIFAR100 images

In our exploratory analysis for the average amount of blue in each image of the CIFAR100 dataset, we found that the data was skewed left with most data clustered around 100. As a result, it can be concluded that most images have less blue than the other main colors.

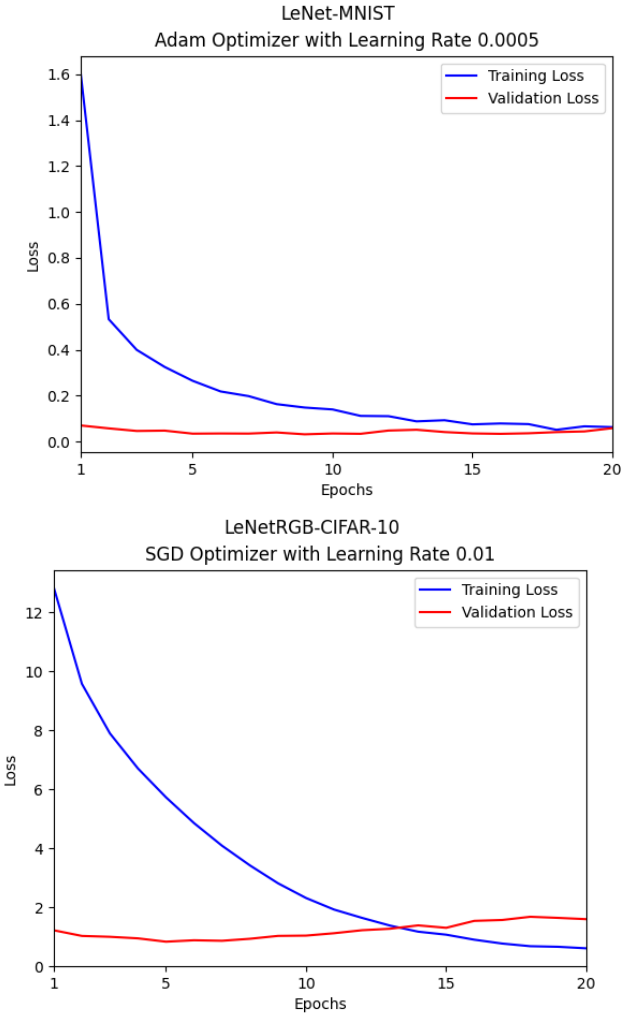




Distribution of average amount of blue in all CIFAR100 images

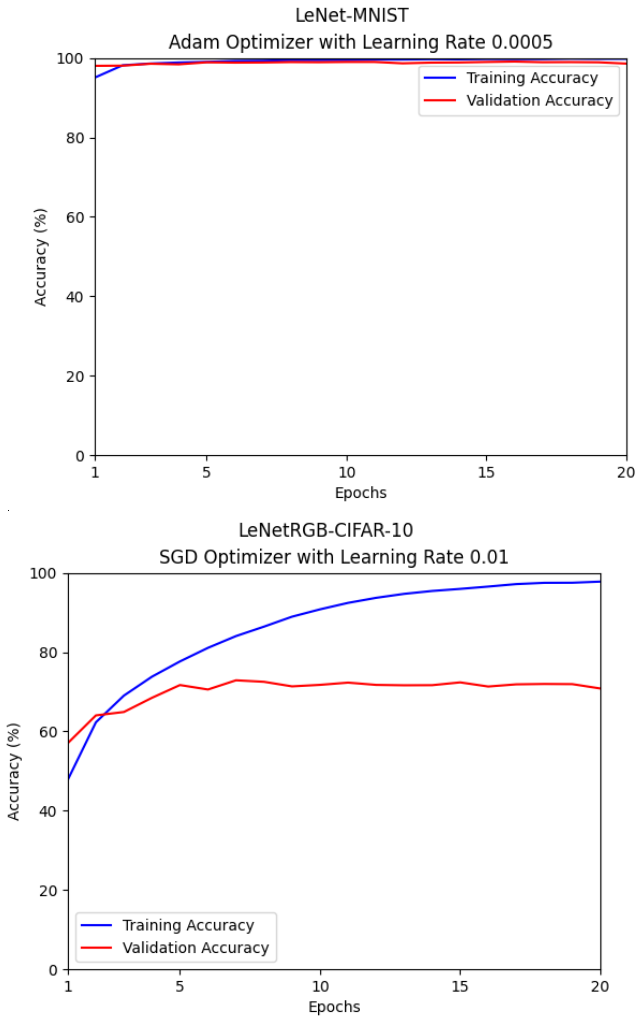
In our exploratory analysis for the average amount of blue in each image of the CIFAR100 dataset, we found that the data was skewed left with most data clustered around 100. As a result, it can be concluded that most images have less blue than the other main colors.

B. LeNet vs. LeNet_RGB



Loss graph of LeNet with MNIST and LeNet_RGB with CIFAR10

In our experiments, we found that with optimal conditions, LeNet converged by loss at roughly epoch 18 and LeNet_RGB converged roughly at epoch 13. Both graphs practically converged (converged by accuracy) at a much earlier point.

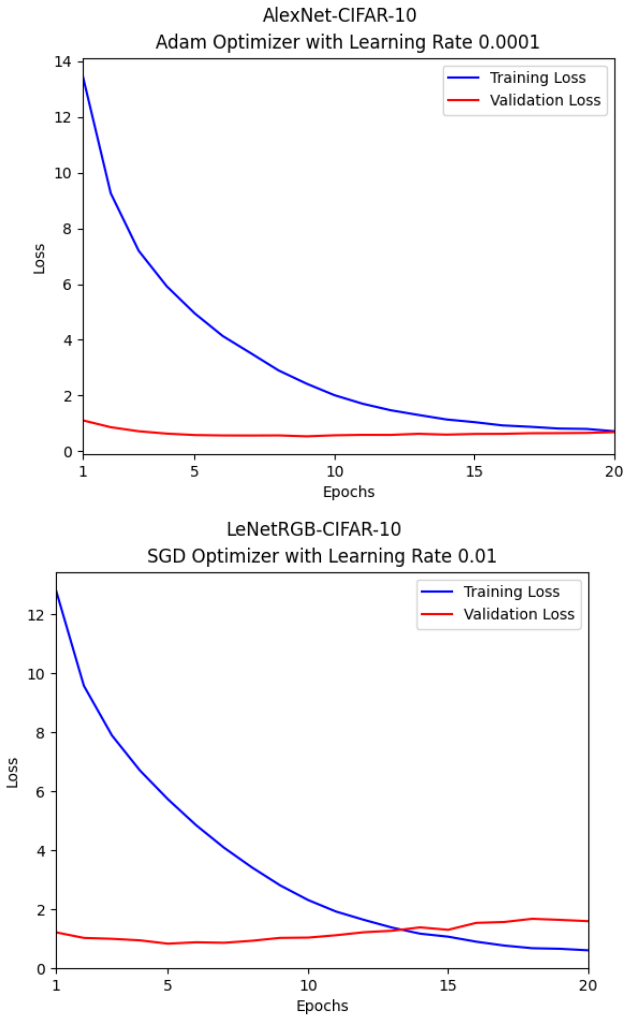


Accuracy graph of LeNet with MNIST and LeNet_RGB with CIFAR10

The original LeNet practically converged at 5 epochs and LeNet_RGB converged at 7 epochs. Following this point, the accuracy experienced very minimal changes. We see a similar behavior between LeNet and LeNet_RGB in the number of epochs to practically converge.

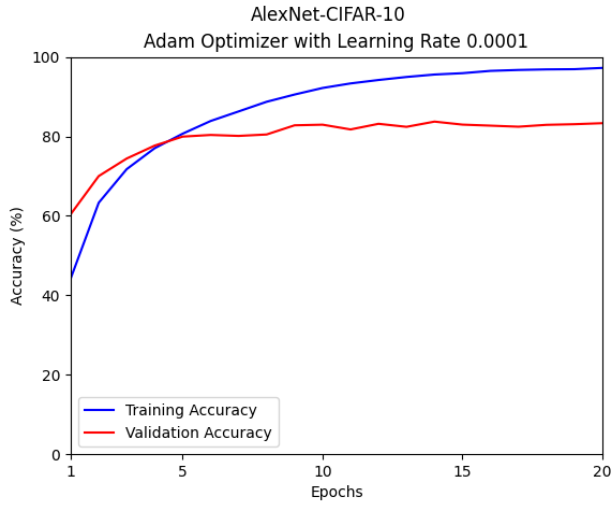
C. LeNet_RGB vs. AlexNet

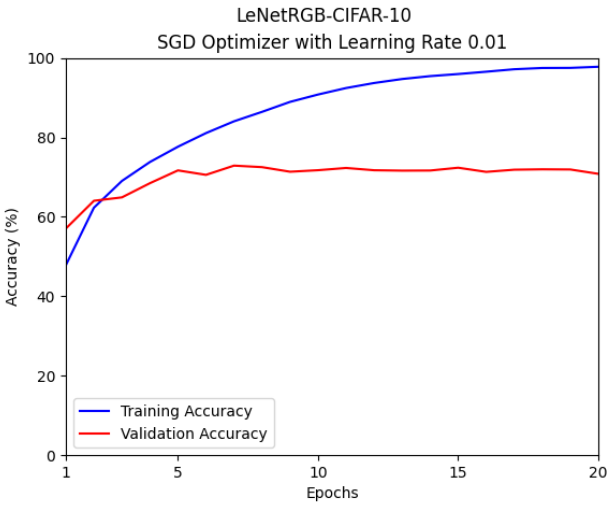
1) CIFAR10 comparisons



Loss graph of AlexNet and LeNet

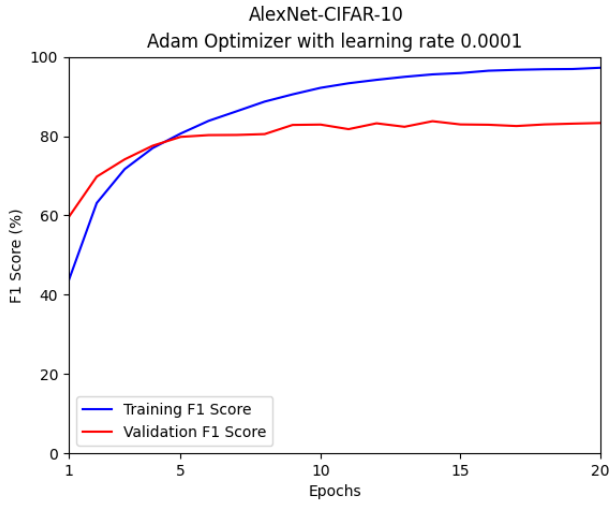
In our experiments, we found that with optimal conditions, AlexNet converged by loss at roughly epoch 20 and LeNet_RGB converged roughly at epoch 13. However, both graphs converged by accuracy much earlier.

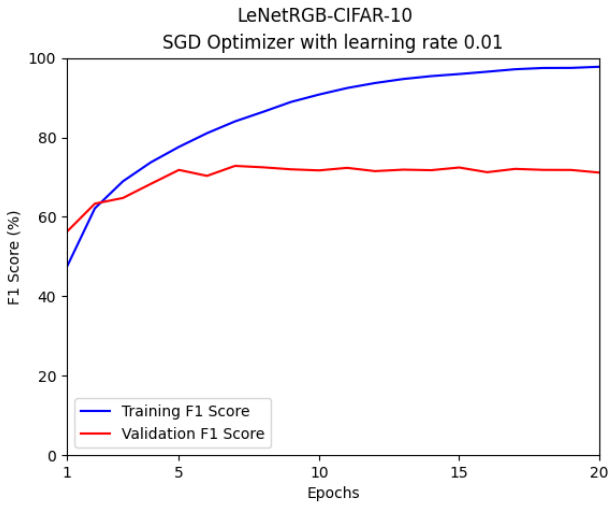




Accuracy graph of AlexNet and LeNet

We found that AlexNet converged by accuracy around epoch 9 and LeNet_RGB converged by accuracy in roughly 7. Following this point, the accuracy practically stagnated. In the AlexNet graph, the difference in time from the practical convergence to the convergence by loss caused an increase in time from 8:42 to 20:20. For LeNet, this was an increase from 4:00 to 8:49. For many students, doubling the execution time for marginal gains is not a worthwhile tradeoff. We believed that if there was a noticeable increase in the F1 scores following this point, there may be a practical educational purpose in running more trials.



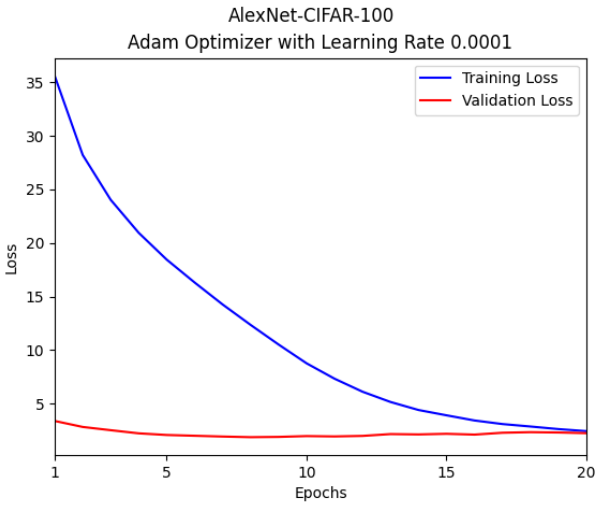


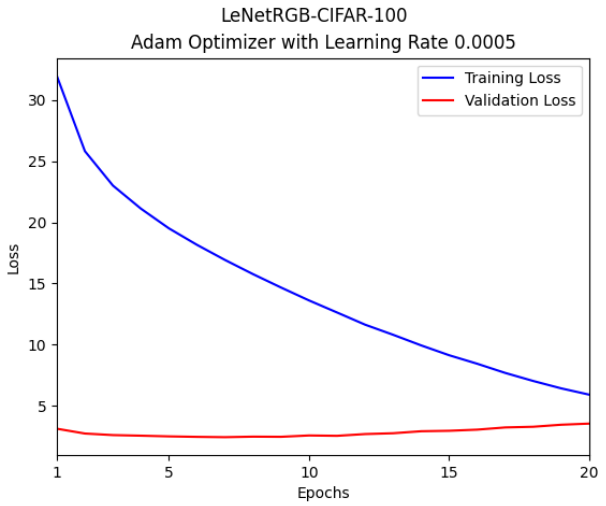
F1 score graph of AlexNet and LeNet

However, we found that the F1 score never fluctuated more than 2% from the practical point of convergence. As a result, when calculating the time consumption for these graphs, we settled on using the practical convergence of AlexNet and LeNet_RGB to calculate the difference in time required. We found that AlexNet required 8:42 and LeNet required 4:00.

As a result, we concluded that AlexNet took over 100% more time than LeNet_RGB to converge. However, there was an overall difference in accuracy of 10%. We feel that this difference is acceptable for the sake of gaining exposure to neural networks and learning concepts to apply for future projects.

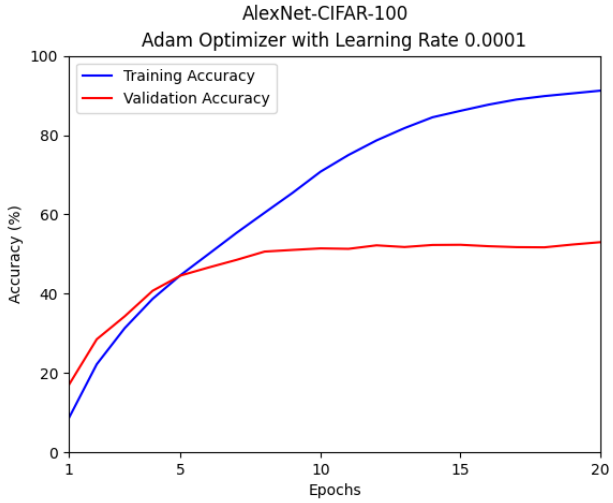
2) CIFAR100 comparisons

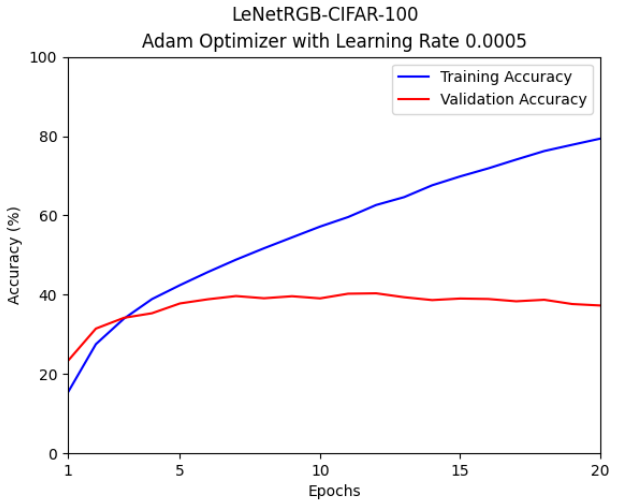




Loss graph of AlexNet and LeNet

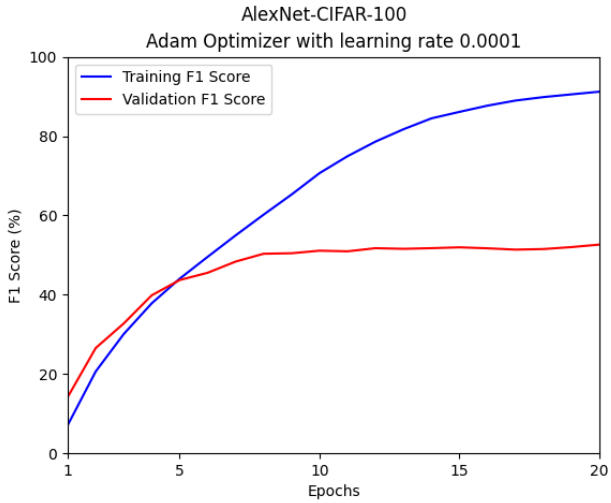
In our experiments, we found that with optimal conditions, AlexNet converged by loss at epoch 20 and LeNet_RGB converged past epoch 20. However, similarly to with CIFAR10, the dataset practically converges much earlier than the real convergence.

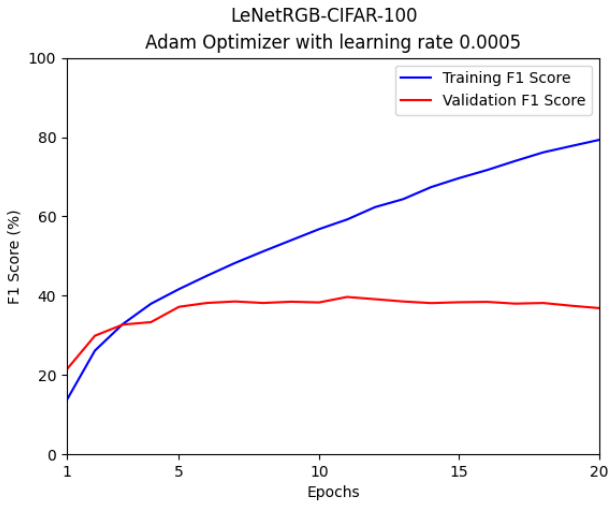




Accuracy graph of AlexNet and LeNet

We found that AlexNet converged by accuracy around epoch 9 and LeNet_RGB converged at epoch 8. AlexNet practically converged at 9:57 and converged by accuracy at 22:04. Meanwhile LeNet_RGB took 5:58 to practically converge and would take over 17:40 to converge by accuracy. Once again, the F1 score did not experience a significant increase following this point.





F1 score graph of AlexNet and LeNet

Because the F1 score never fluctuated more than 2% from the practical point of convergence, we concluded that the graphs had reached convergence for the intended purpose.

With a difference in accuracy of roughly 13% between the two models, there is a significant difference in the overall effectiveness of the 2 algorithms. While the time increase of roughly 28% benefits LeNet_RBG, the difference of accuracy of 13% in AlexNet's favor is a noteworthy improvement in accuracy. We believe that both neural networks would likely require more datapoints for each class to improve on accuracy.

VI. Final results

A. Training data

algorithm	dataset	epoch	accuracy	F1	loss
LeNet	MNIST	15	99.70%	99.70%	0.07502
LeNet_RGB	CIFAR10	9	85.20%	85.19%	3.706
AlexNet	CIFAR10	15	93.32%	93.32%	1.753
LeNet_RGB	CIFAR100	11	59.59%	59.23%	12.62
AlexNet	CIFAR100	19	90.53%	90.53%	2.639

B. Validation data

algorithm	dataset	epoch	accuracy	F1	loss
LeNet	MNIST	15	99.08%	99.08%	0.03552
LeNet_RGB	CIFAR10	9	72.28%	72.17%	0.8965
AlexNet	CIFAR10	15	82.09%	82.04%	0.6168
LeNet_RGB	CIFAR100	11	40.25%	39.69%	2.525
AlexNet	CIFAR100	19	52.4%	52.01%	2.321

C. Test data

algorithm	dataset	epoch	accuracy	F1	loss
LeNet	MNIST	15	99.20%	99.20%	0.0289
LeNet_RGB	CIFAR10	9	72.89%	72.72%	0.8924
AlexNet	CIFAR10	15	80.64%	80.74%	0.7296
LeNet_RGB	CIFAR100	11	39.28%	38.33%	2.7138
AlexNet	CIFAR100	19	45.86%	45.94%	3.1556

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems* (F. Pereira, C. Burges, L. Bottou, and K. Weinberger, eds.), vol. 25, Curran Associates, Inc., 2012.

[2] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[3] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[5] S. Gross, "Alexnet.py." https://github.com/pytorch/vision/blob/main/torchvision/models/alexnet.py, 2017.

[6] Nouman, "Writing lenet5 from scratch in pytorch," 2022.

[7] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, *Dive into Deep Learning*. Cambridge University Press, 2023. https://D2L.ai.

[1] [2] [3] [4] [5] [6] [7]