**Image Classification with Pytorch Deep Learning Framework**

Yeray Lopez

***Abstract***–**Neural networks and deep learning have become an increasingly hot topic in the world of Artificial Intelligence. Hardware is now more accessible than ever through online tools that provide free access to computational resources such as CPUs, GPUs, and even the famed TPUs use for specialized workloads at Google. We look at one of these free services and run them with GPU hardware acceleration to determine the best deep learning model for image classification on the CIFAR-10 dataset based on a variety of epochs.**

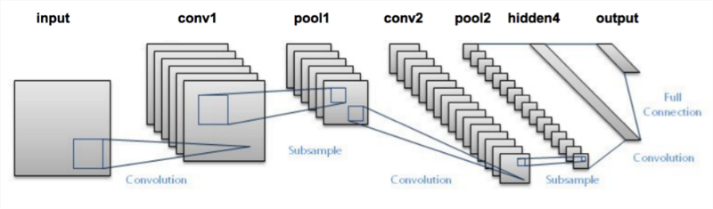
I. INTRODUCTION

The growing demand for machine learning and deep learning is continuing to rise and propagate throughout industries. Due to this demand, new cloud computing resources have also been heavily sought after to the point where companies can offer free use of cloud computational resources such as storage, RAM, CPUs, and GPUs. These companies’ services include Kaggle Kernel, Jupyter Notebook on Google Cloud Platform, Google Colaboratory, Amazon SageMaker, and Azure Notebooks. Unfortunately, many students and people in the data science industry are unaware about these resources and miss out on a crucial opportunity.

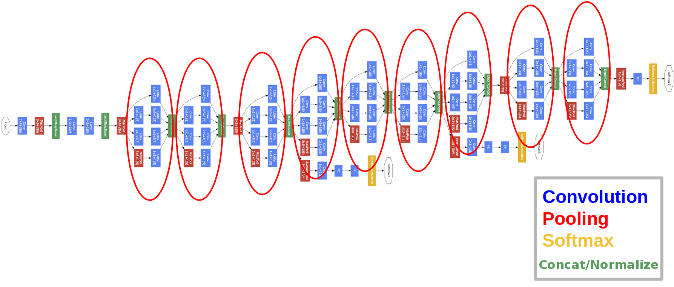
In this assignment, we explore one of these services: Google Colaboratory (Colab). Google Colab, which was originally a research project that was under Jupyter, has now been integrated into Google after they collaborated on the project. Colab is an extremely powerful tool as a cloud notebook because the environment is set up for you and also comes preinstalled with over 300+ data science and machine learning libraries. Colab also gives us the option to use an NVIDIA Tesla K80 GPU or Google TPUv2. Both of which can provide a much faster training time compared to a CPU. In this assignment, we will only be using the GPU due to its obvious advantages and time saving benefits that it has over the standard issued CPU. Our work will be on image classification on the CIFAR-10 dataset.[1] We will utilize Colab with the deep learning framework PyTorch, which was created by Facebook AI. It offers a very powerful but simple to program deep learning framework. We will be looking at LeNet, GoogLeNet and ResNet18 to compare the performance of these neural networks to each other.

II. RELATED WORK

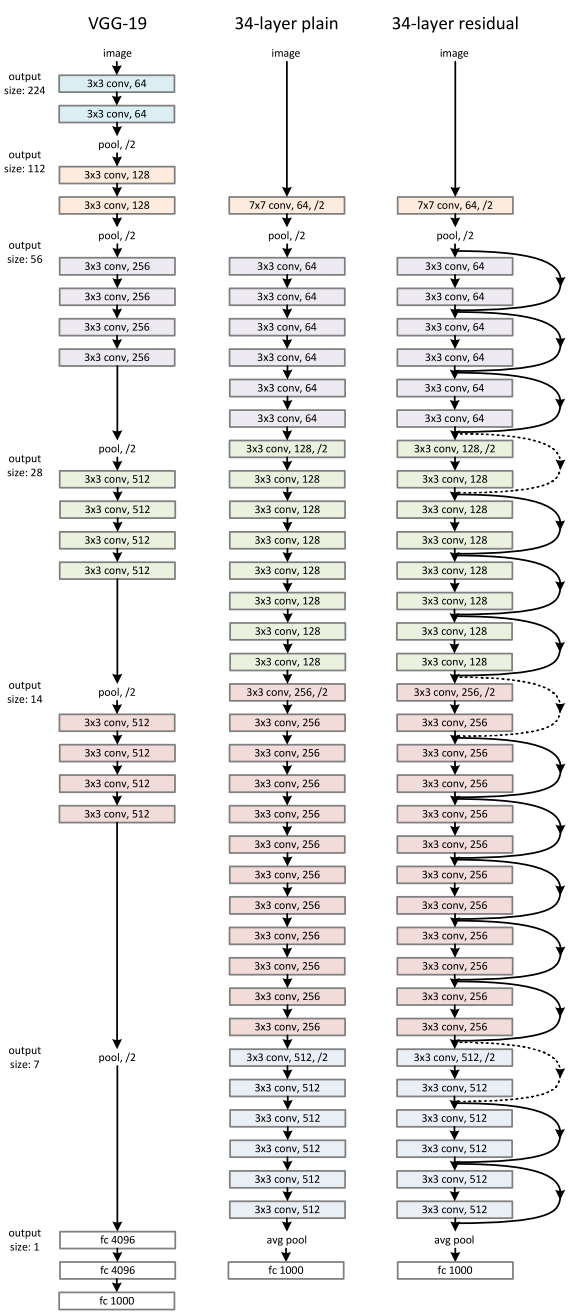
Before we dive into the implementation, the models in question will be briefly discussed. LeNet was primarily used for OCR and character recognition in documents when its’ architecture was first conceived. As seen in this assignment, we know LeNet can be used for image classification as well. It can run off a CPU or GPU, which makes it a great option as a first CNN with 5 layers to allow for quick training times.



The next model addressed in this assignment is GoogLeNet, which has 22 layers and is mostly used for computer vision, image classification, and object detection. GoogLeNet uses 12 fewer parameters compared to AlexNet, making it faster and more accurate. The expanded architecture can be found in the appendix.



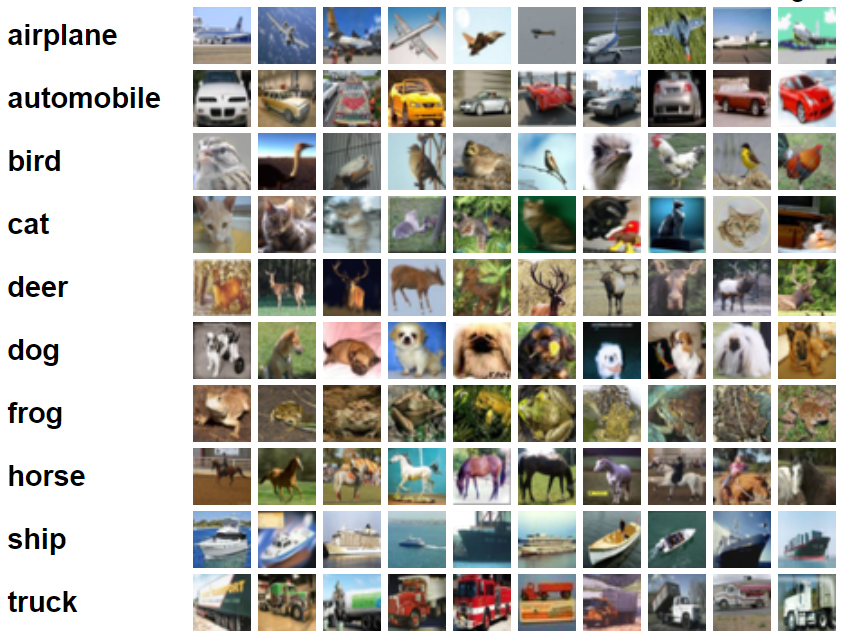
Last, but not least, we have ResNet, which is a fascinating model because it goes against the intuition that adding more layers means better results. Most of the time, adding layers to a NN brings forth better results but sometimes it can have the opposite effects due to the vanishing gradient problem. Where the model weights of the first layers can’t be updated correctly through the backpropagation of the error gradient (the chain rule multiplies error gradient values lower than one and then, when the gradient error comes to the first layers, its value goes to zero). ResNet solves this issue by utilizing the identity matrix and preserving the gradient and avoiding unnecessary multiplication operations. For the purpose of this assignment, ResNet-16 will be used.



III. DATA ANALYSIS

We are working with the CIFAR-10 dataset. It consists of 60000 32x32 color images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. The dataset is divided into five training batches and one test batch, each with 10000 images. The test batch contains exactly 1000 randomly-selected images from each class. The training batches contain the remaining images in random order, but some training batches may contain more images from one class than another. Between them, the training batches contain exactly 5000 images from each class.[1]

Here are the classes in the dataset, as well as 10 random images from each:



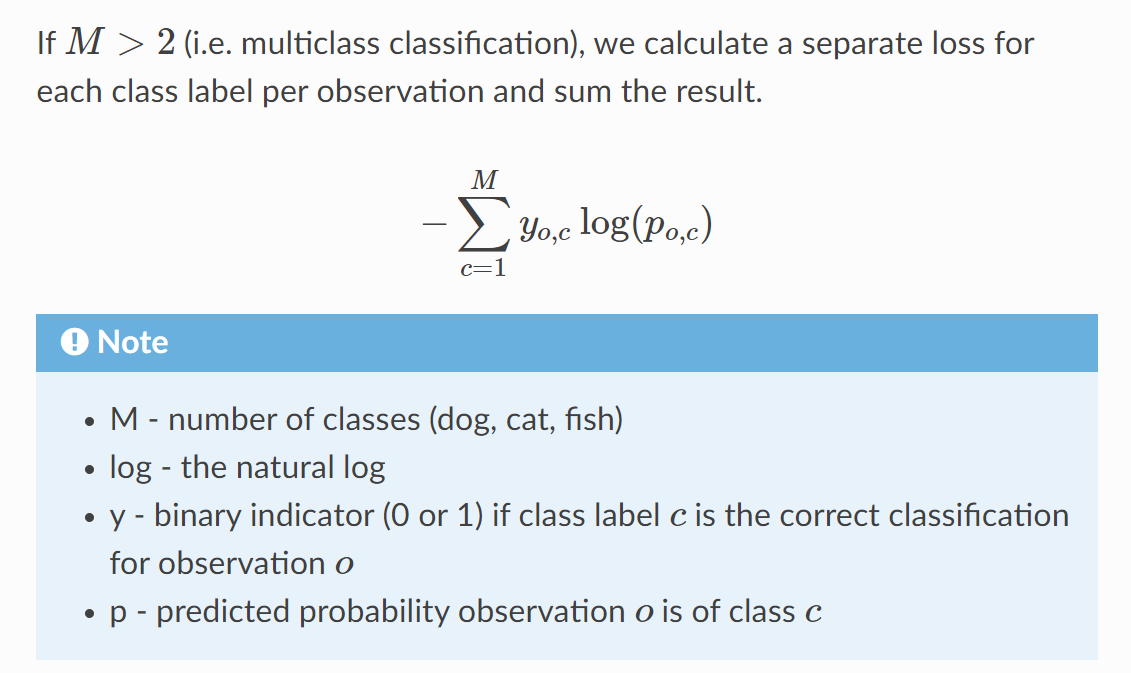
IV. PREDICTION MODEL

No preprocessing is required for the dataset so we can directly work with it.

1. *Evaluation*

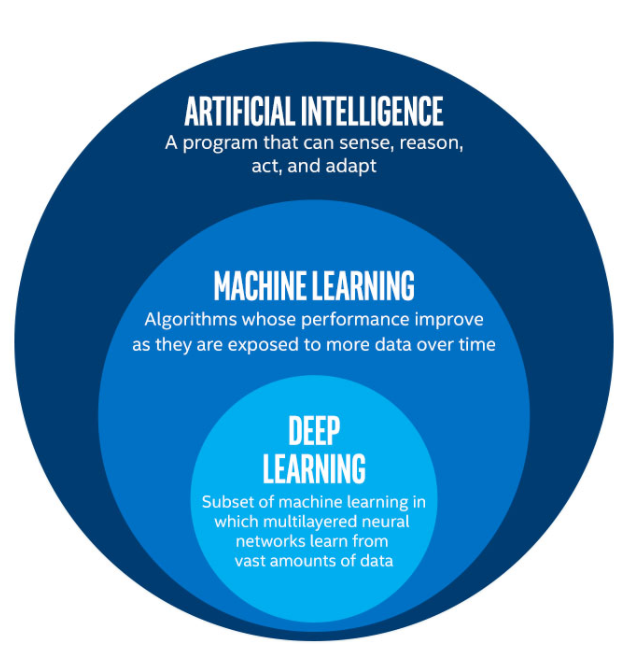
Accuracy is used as our primary evaluation scheme. We will determine the accuracy for each class and average them for the overall accuracy of the model.

The Loss function is also used to evaluate the performance of the classification model. In this case, we are using Cross-entropy loss, or log loss.



1. *Deep Learning*

Deep Learning is a subset of machine learning within AI. It “imitates” the human brain by processing data and creating patterns. Deep learning is capable of learning both supervised and unsupervised. Deep learning utilizes neural networks which loosely resemble neurons in the human brain and connect like a hierarchical web. Compared to logical or linear approaches utilized in traditional machine learning methods, deep learning implements nonlinear techniques and uses them to solve tasks in a fraction of the time that it would take a linear or logical approach. In this paper, we will focus on a class of deep learning models called Convolutional Neural Network (CNN). [2]

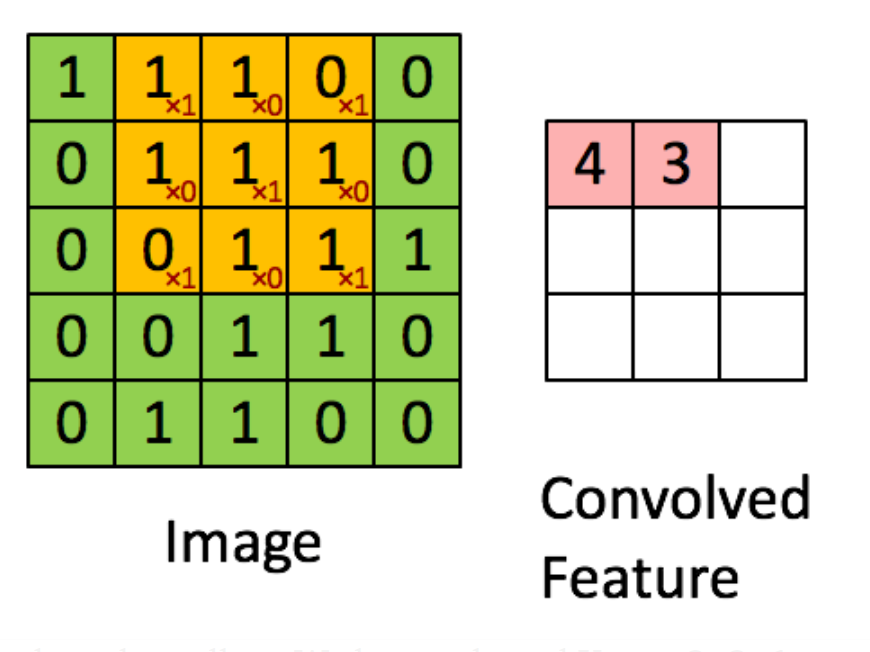
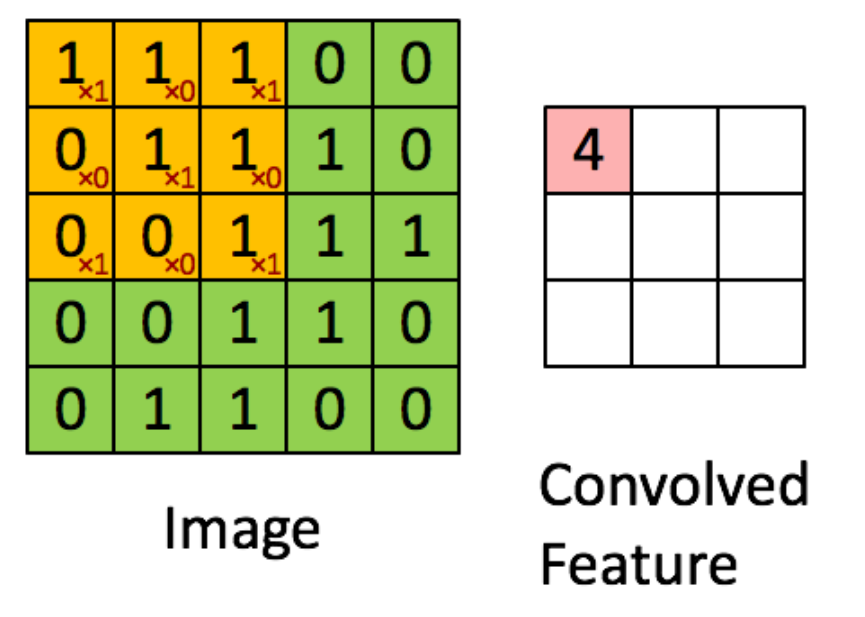


* 1. *Convolutional Neural Networks (CNN)*

A Convolutional Neural Network (CNN or ConvNet) are a special kind of multi-layer neural networks designed to recognize visual patterns directly from pixel images with minimal preprocessing. All of the models used in this assignment fall under CNNs. Some of the basic layer operations will be covered in the following sections. [3]

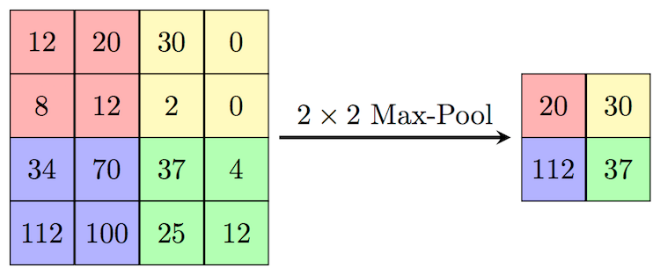
* + 1. *Convolution*

A convolution is the simple application of a filter to an input that results in an activation. Repeated application of the same filter to an input returns a map of activations called a feature map, indicating the locations and strength of a detected feature in an input, such as an image. [4]



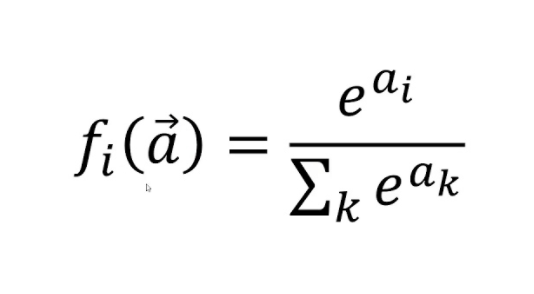
* + 1. *Pooling*

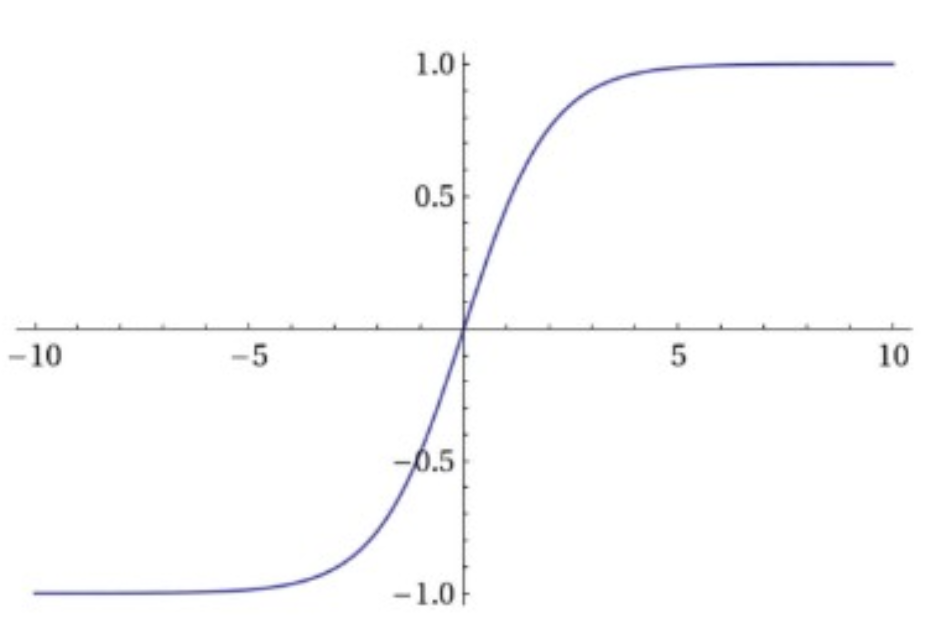
Pooling is a function to progressively reduce the spatial size of the representation to decrease the number of parameters and computation in the network. Pooling layer operates on each feature map independently. Typically pooling is seen in CNNs as max pooling.



* + 1. *SoftMax*

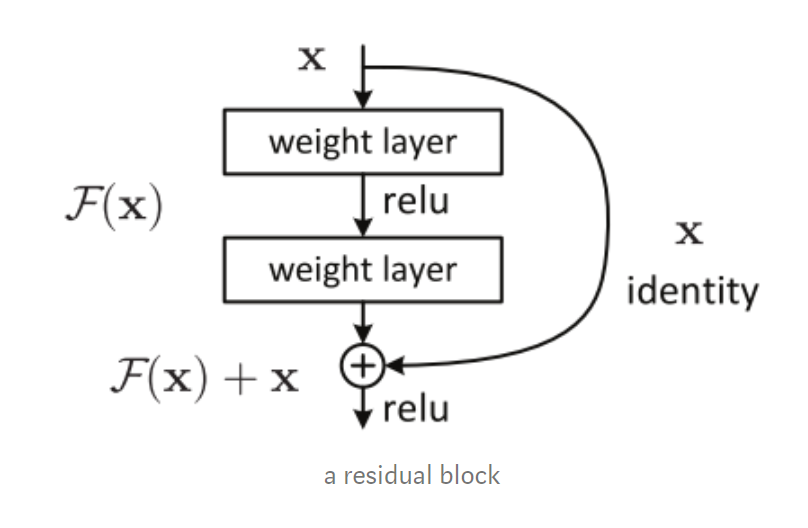
The SoftMax regression is a form of logistic regression that normalizes an input value into a vector of values that follows a probability distribution whose total sums up to 1. The output values are between the range [0,1] and we are able to avoid binary classification and accommodate as many classes or dimensions in our neural network model. This is why SoftMax is sometimes referred to as a multinomial logistic regression.





* 1. *Residual Neural Network (ResNet)*

Residual Neural Network is a neural network that skips layers using “identity shortcuts”. This is done to avoid the vanishing gradients problem and shorten training times. The main importance of a ResNet is its ability to skip layers.



Now that the basic building blocks of how neural networks function, we can evaluate their performance.

V. EVALUATION

In this section we evaluate and compare the performance of different CNNs. We present our findings and view the results in terms of accuracy at different epochs and loss values throughout the training process.

All models were trained at:

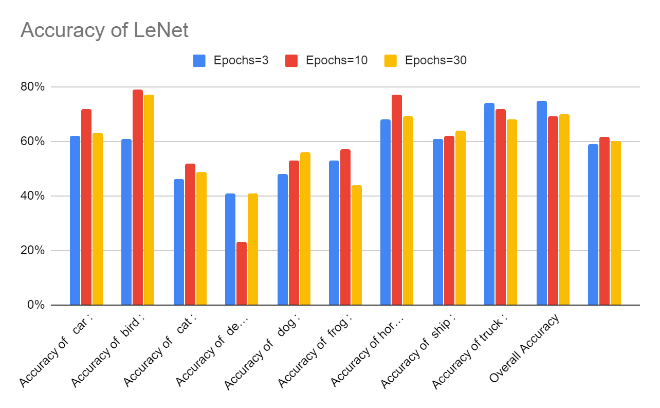
***Learning Rate = 0.001 and Momentum = 0.9***

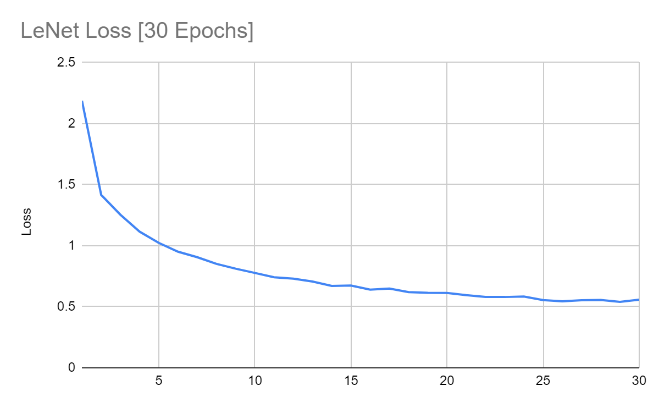
***Loss Function: Cross Entropy***

Epochs were arbitrarily chosen and longer training was forgoed for GoogLeNet and ResNet due to their extensive training times despite using GPU acceleration.

1. *LeNet*

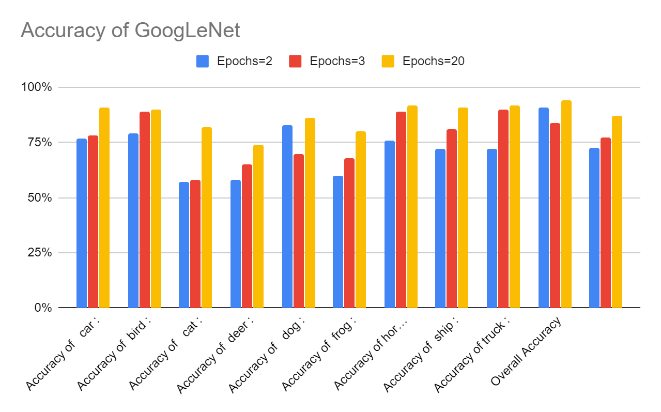
Starting with LeNet, the smallest of the CNNs being only 5 layers deep, we peak at 62% overall accuracy after training for 10 epochs. The performance actually diminishes at 30 epochs due to overtraining the model. This is an important thing to note as overtraining a model can negatively impact your results. Loss never truly gets too close to zero so given that fact that it was trained within 30 epochs, our best results would lie in the same range as the minima, between 25 and 30 epochs.

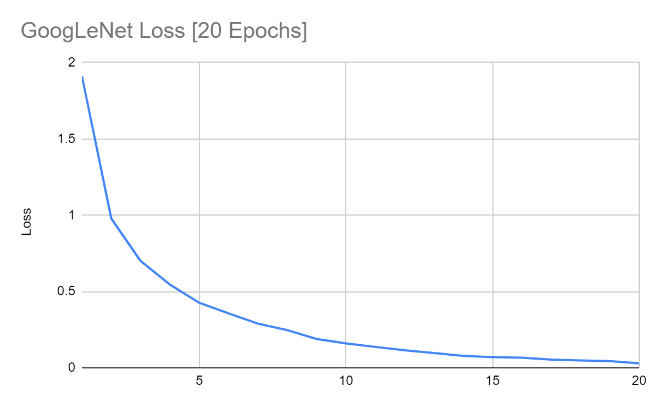




1. *GoogLeNet*

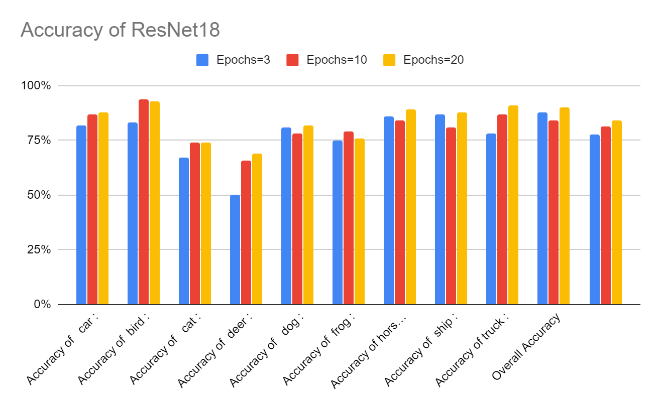
Moving onto GoogLeNet which is the largest CNN of the three CNNs,sitting at 22 layers. GooLeNet’s accuracy peaks at 87% at epoch 30 with a loss function that continues to approach zero. We only stopped at 30 epochs due to time constraints, but better results can definitely be achieved with more training.

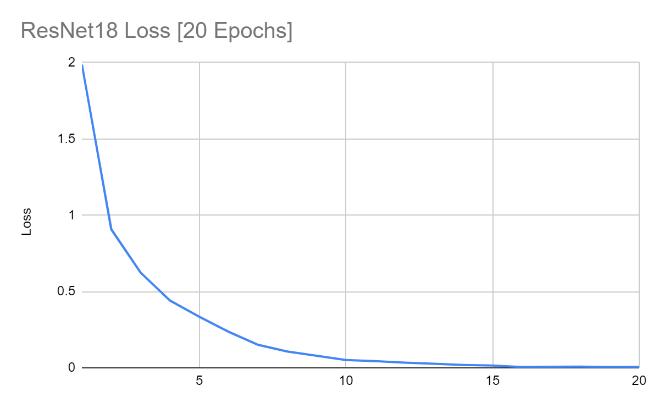




1. *ResNet18*

ResNet18 displays results very similar to GoogLeNet in terms of overall accuracy but it significantly differs in behavior in terms of individual classifications. There are some similarities and differences that can be attributed to what features it is detecting inside the neural network. ResNet18 was also significantly faster with only 18 layers and its layer skipping properties.





VI. CONCLUSION

In this report we used various CNNs to perform image classification on the CIFAR-10 dataset. There were 10 different classes that were tested for accuracy and overall accuracy was calculated by averaging all accuracies. When comparing all 3 models, GoogLeNet is the best in terms of accuracy. However, ResNet is much more efficient to train if a time constraint is a factor. This goes to show that different neural networks have different advantages over each other. It’s important to recognize that this is only one specific dataset from the myriad that exist. Models may outperform each other completely differently on other datasets. We utilized Colab and GPU acceleration in PyTorch to aid in significantly reducing the training time.

References

1. The CIFAR-10 dataset <https://www.cs.toronto.edu/~kriz/cifar.html>
2. Medium <https://medium.com/analytics-vidhya/cnns-architectures-lenet-alexnet-vgg-googlenet-resnet-and-more-666091488df5>
3. Towards Data Science <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>
4. Machine Learning Mastery <https://machinelearningmastery.com/convolutional-layers-for-deep-learning-neural-networks/>