Introduction To Deep Learning With Image Classification

**Lee Hao Wei**

**ABSTRACT**

Recent research in deep learning has been largely inspired by the way our brain works. When you think of it, it is fascinating to know that with a given input, our brain processes features that say let us know of the world that surrounds us. In this paper we will use the ever so popular MNIST dataset to explore different deep learning architectures and create models of our own. Our objective is to try to create a deep learning model which can correct identify the classes in MNIST

# Introduction

In this paper, we will attempt to create deep learning models using TensorFlow that will be able to correctly classify the images in the MNIST dataset. We will go in depth to each architectures use. Hopefully after reading you will be able to have a deeper understanding of the deep learning process.

# Background

## Deep Learning

Deep Learning is a subfield of machine learning concerned with algorithms inspired by the structure and function of the brain called artificial neural networks. [2] One of the benefits of Deep Learning is the ability to learn massive amounts of data compared to Machine Learning. On top of that, it is proven that Deep Learning has outperformed popular Machine Learning techniques in many domains, e.g., natural language processing, image classification, and cybersecurity.

## Image Classification

Image classification is where a computer can analyze an image and identify the ‘class’ the image falls under. (Or a probability of the image being part of a ‘class’.) A class is essentially a label, for instance, ‘car’, ‘animal’, ‘building’ and so on. Early image classification relied on raw pixel data. This meant that computers would break down images into individual pixels. The problem is that two pictures of the same thing can look very different. They can have different backgrounds, angles, poses, etcetera. This made it quite the challenge for computers to correctly ‘see’ and categorize images.[1] Deep learning helps solve that issue. We can use CNN to help classify images better. We will go more in-depth about CNNs later.

# Related works

As of the time of writing, there have been several works on deep learning and how it works. There are papers on CNN, RNN, LSTM and different techniques involved in deep learning. One paper published by Springer Open explains and describes the history of Deep Learning, popular architectures, challenges faced and applications. Demonstration just how versatile Deep Learning is.[3]

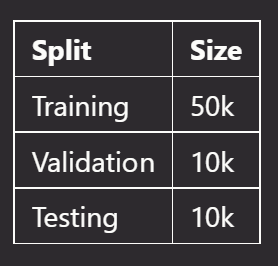
# Methodology

## Data Set

The data set used in this paper is MNIST. MNIST set is a large collection of handwritten digits. It is a very popular dataset in the field of image processing. It is often used for benchmarking machine learning algorithms. MNIST contains a collection of 70,000, 28 x 28 images of handwritten digits from 0 to 9. [7]

## Loading the Dataset

We will be using TensorFlow to load MNIST. TensorFlow is a python library created by google that ease the process of acquiring data, training models, serving predictions, and refining future results.[4] When loading the dataset, we will want to split our dataset into 3 sections. The 3 sections are training data, which will be used to train our model, test data, which is used to provide an unbiased evaluation of a final model fit on the training dataset and lastly validation data, which is used to give an estimate of model skill while tuning model’s hyperparameters.[5] The split of the dataset is show in Fig 1. Fig 1



## Exploratory Data Analysis

Before we start clustering, we must first perform exploratory data analysis. Exploratory Data Analysis (EDA) is the crucial process of using summary statistics and graphical representations to perform preliminary investigations on data in order to uncover patterns, detect anomalies, test hypotheses, and verify assumptions.[6] Some basic findings are that the images are in the same of 28 by 28 pixels. The 10 classes are relatively distributed equally as seen from the bar plot in Fig 2.

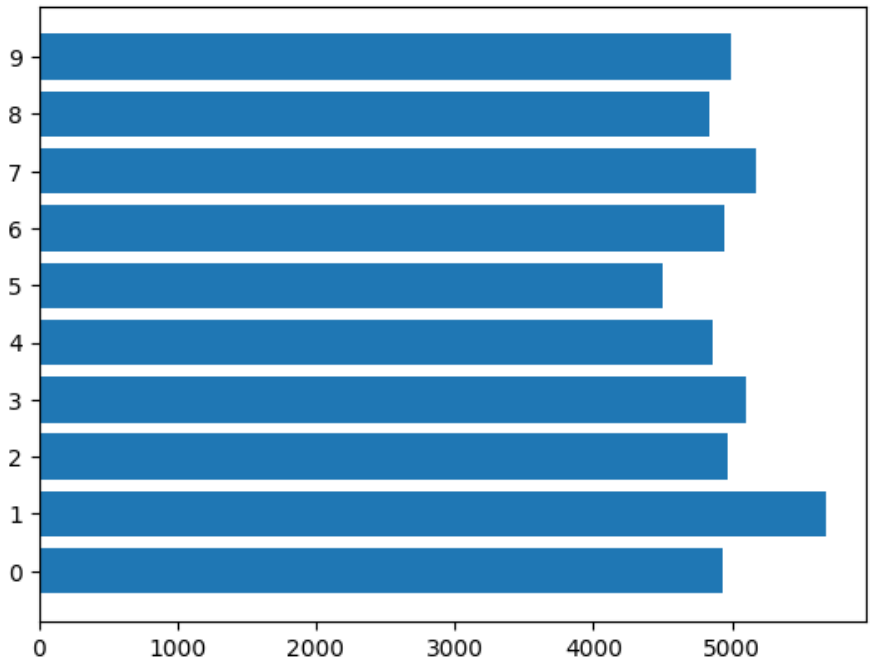


Fig 2

## Data Preprocessing

After performing our exploratory data analysis, we will need to process our data. We will be applying the following steps to our data

1. Normalization

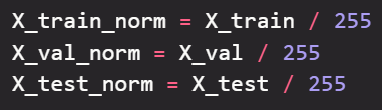
2. Encoding target labels

3. Applying basic data augmentation

4. Create a separate data set with CutMix data augmentation

1. Normalization

**Normalization is a data preparation technique that is frequently used in machine learning. The process of transforming the columns in a dataset to the same scale is referred to as normalization.[7] The reason we do both of those things for image classification is because in the process of training our network, we're going to be multiplying ‘weights’ and adding to ‘biases’ these initial inputs to cause activations that we then backpropagate with the gradients to train the model. We'd like in this process for each feature to have a similar range so that our gradients don't go out of control. Another way of thinking about it, is deep learning networks traditionally share many parameters - if you didn't scale your inputs in a way that resulted in similarly ranged feature values sharing wouldn't happen very easily because to one part of the image ‘weight’ is a lot and to another it's too small. The way we will implement Normalization to our dataset is by dividing all values in our dataset by 255 as shown in Fig 3. This is because 255 is the maximum value, dividing by 255 expresses a 0-1 representation.**



**Fig 3**

1. **Encoding Target Labels**

To encode our target labels, we will be using a method from TensorFlow called ‘to\_categorical’. Using the method ‘to\_categorical’, a NumPy array (or) a vector which has integers that represent different categories, can be converted into a NumPy array (or) a matrix which has binary values and has columns equal to the number of categories in the data.[8]

1. Basic Data Augmentation

Data augmentation is a set of techniques to artificially increase the amount of data by generating new data points from existing data. This includes making small changes to data or using deep learning models to generate new data points.[9] In fear of our model overfitting, I used simple data augmentation techniques. Since our dataset consist of numbers, we cannot apply more heavy data augmentation such as flipping the image as numbers like 6 and 9 would 100% be wrongly classified. We will be using a method from TensorFlow called ‘ImageDataGenerator’ to assist us in this process. Fig 4 shows the data augmentation parameters used.

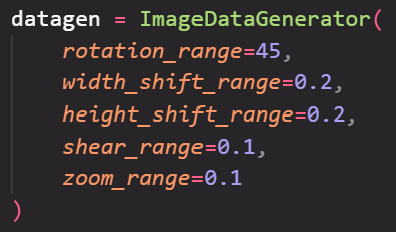


Fig 4

1. CutMix Data Augmentation

As we know that to improve the performance of an ML model, one of the ways is to implement Data Augmentation in our data pre-processing steps. There are various ways of Data Augmentation, one such technique introduced recently is CutMix. In CutMix augmentation we cut and paste random patches between the training images. The ground truth labels are mixed in proportion to the area of patches in the images. CutMix increases localization ability by making the model to focus on less discriminative parts of the object being classified and hence is also well suited for tasks like object detection.[10]

# Experiment

## Modelling

Using data with no data augmentation, data with basic data augmentation and data with CutMix data augmentation, we will attempt to create a deep learning model which can accurately classify each image in our dataset. We will be using validation accuracy as the metric to see how well our model performs.

## Baseline Model

A baseline model is essentially a simple model that acts as a reference in a machine learning project. Its main function is to contextualize the results of trained models. Baseline models usually lack complexity and may have little predictive power.[11] For our case we will be using fully connect neural networks as our baseline model. Our baseline model performs the best with data with no augmentation, with a validation accuracy of 98.48% seen in Fig 5. The accuracy of our baseline model is high, this is because the complexity of our dataset is low, and it is easy for model to accurately predict the classes

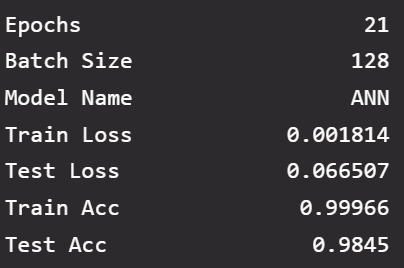


Fig 5

## VGG16

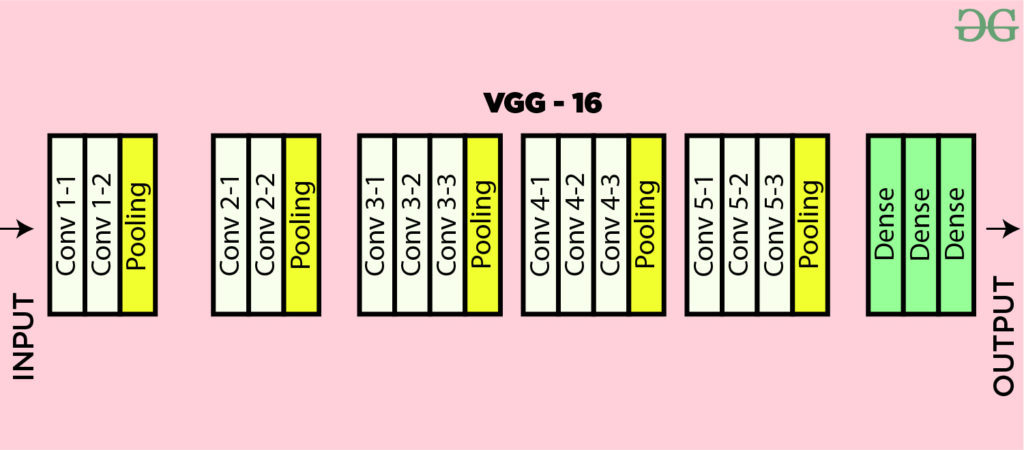


Fig 6

An article by medium describes VVG16 as “a type of CNN (Convolutional Neural Network) that is considered to be one of the best computer vision models to date. The creators of this model evaluated the networks and increased the depth using an architecture with very small (3 × 3) convolution filters, which showed a significant improvement on the prior-art configurations. They pushed the depth to 16–19 weight layers making it approx — 138 trainable parameters.”.[12] You might be confused as to what is the difference between convolution neural networks and artificial neural networks. To put it simply, CNN tends to be a more powerful and accurate way of solving classification problems compared to ANN. For our implementation of VGG16, I added L2 regularisation on every neural network layer and added batch normalisation after each pooling layer, this helps with managing overfitting. Our dataset performed the best with basic data augmentation on VGG16, achieving a result of 98.81% as shown in Fig 7.



Fig 7

## ResNet50

ResNet stands for Residual Network and is a specific type of convolutional neural network (CNN) introduced in the 2015 paper “Deep Residual Learning for Image Recognition” by He Kaiming, Zhang Xiangyu, Ren Shaoqing, and Sun Jian. CNNs are commonly used to power computer vision applications.

ResNet-50 is a 50-layer convolutional neural network (48 convolutional layers, one MaxPool layer, and one average pool layer). Residual neural networks are a type of artificial neural network (ANN) that forms networks by stacking residual blocks.

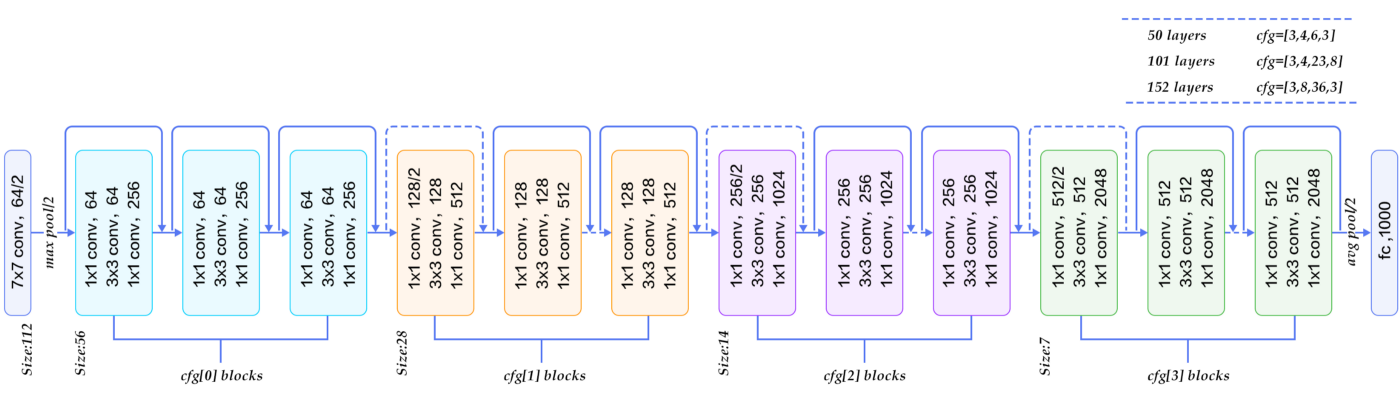


Fig 8

Our dataset performed the best with basic data augmentation on ResNet50, achieving a result of 99.33% as shown in Fig 9. This the best result we had so far. It is probably attributed to the fact that ResNet50 is such a wide model, 50 layers.

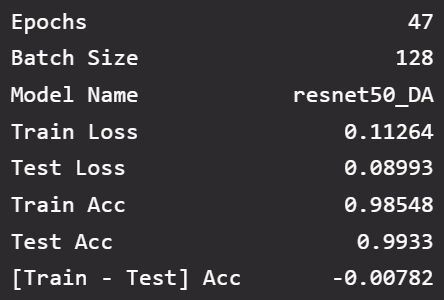


Fig 9

# Model improvement

For model improvement we will be trying to improve our ResNet50 model. There are multiple ways to improve a model. For example, we could add dropout layers to decrease overfitting even more, or we can use squeeze excite blocks. Squeeze-and-Excitation Networks (SENets) introduce a building block for CNNs that improves channel interdependencies at almost no computational cost. They can be easily added to existing architectures.[14] However, since our model already has pretty high validation accuracy we will try simpler model improvement techniques such as hyperparameter tuning the learning rate and weight decay of L2 regularisation.

For our hyperparameter tuning with will be making use of Keras Tuner. We will use a method call ‘Random Search’. ‘Random Search’ only test a certain number of combinations that are selected randomly.[15] This the hyperparameter tuning process much faster as it doesn’t test every combination of hyperparameters.

For our case, hyperparameter tuning did not improve our model. This could be due to use of ‘Random Search’ as it doesn’t test every combination of hyperparameters.

# Model Evaluation

After evaluating the various models and architectures we choose ResNet50 with basic data augmentation as our model as it gave the best validation accuracy out of all the models and combinations tested as shown in Fig 10.

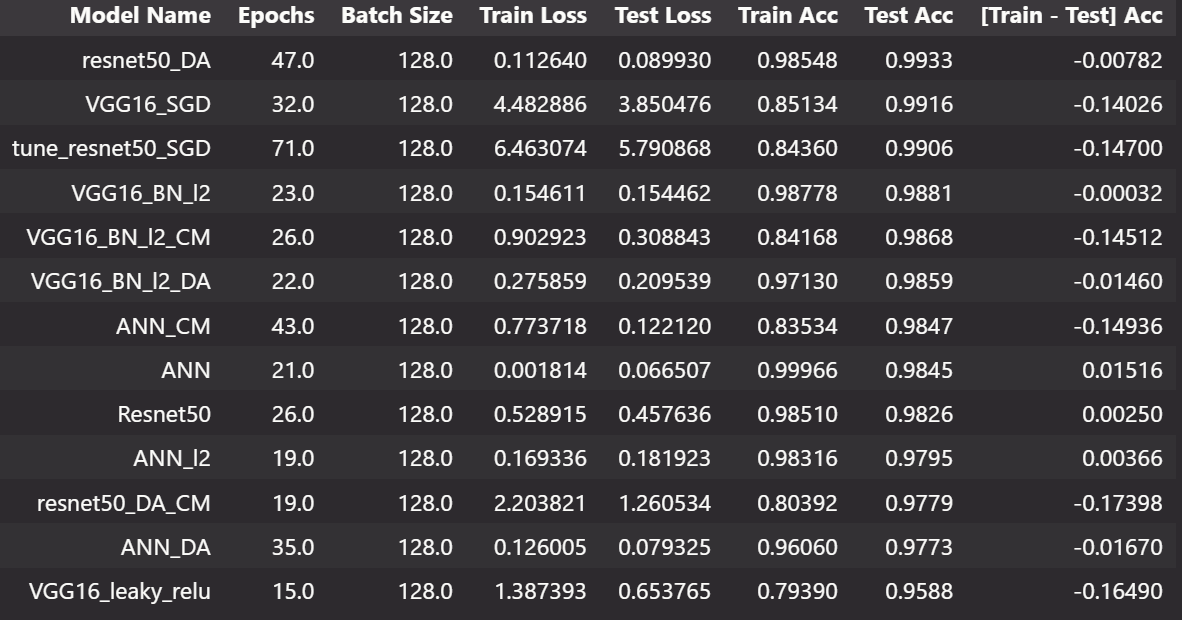


Fig 10

## Evaluation

Now is the time to evaluate my final model. To ensure it generalizes well, I want to ensure a accuracy on the testing set consistent with that on the validation set. We will use a normalised version of our test set as we trained the model with normalised data. Our final accuracy on the test set is 98.91%. This is extremely close to the validation set results, suggesting that our model does not overfit and can generalize well without issues.

## Classification Report

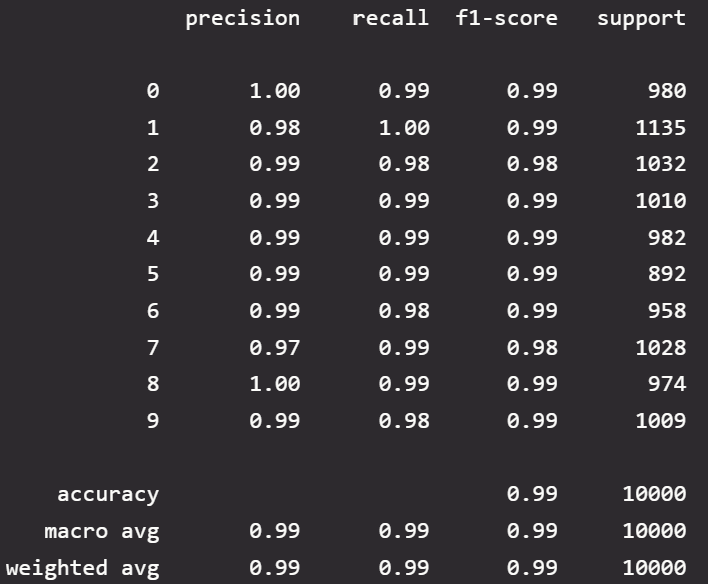


Fig 11

From Fig 11, we can see that our model doesn’t have a particular class that is underperforms. It performs about the same when predict each of the classes

## Error Analysis

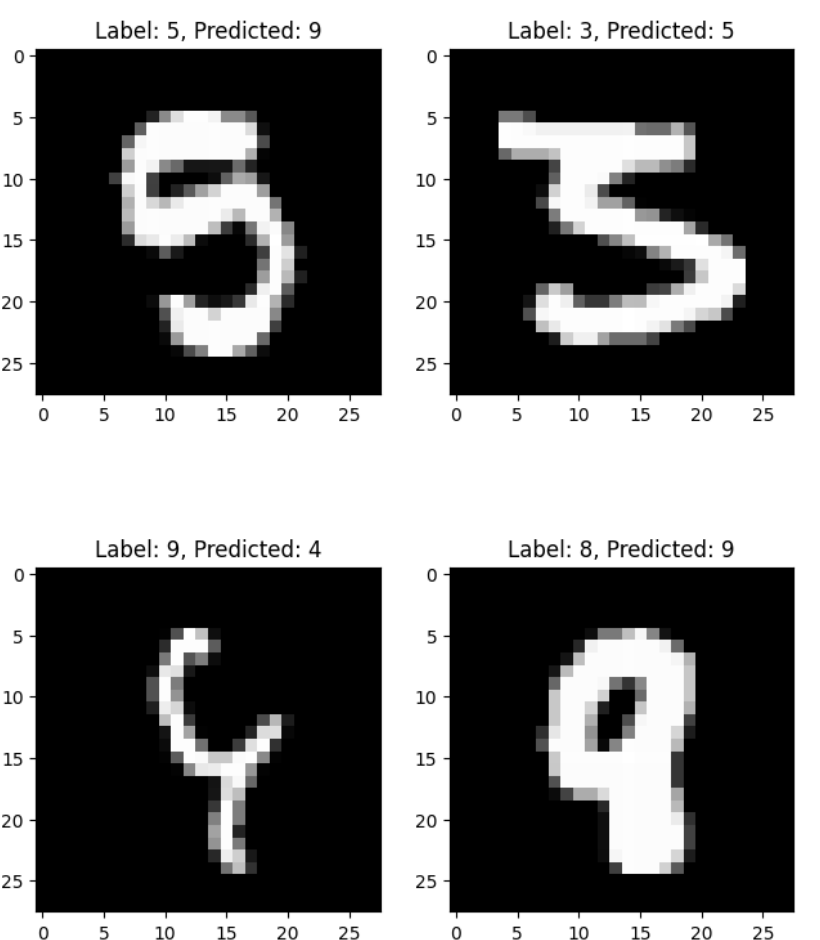


Fig 12

For error analysis we will plot a few images that our model did not predict successfully. We can see from Fig 12 that the images that our model predict incorrectly are reasonable. In the end it is hard to get an accuracy of 100% even on this simple of a dataset.

# CONCLUSION

In conclusion, we successfully made a deep learning model using convolution layers and experiment with different types of architecture. We went through the processes needed when attempting to create a deep learning model. I hope you found this paper beneficial.

# References

1. Reed, N. (2020, December 3). ELI5: What is image classification in deep learning? ThinkAutomation. Retrieved November 24, 2022, from <https://www.thinkautomation.com/eli5/eli5-what-is-image-classification-in-deep-learning/>
2. Brownlee, J. (2020, August 14). What is deep learning? MachineLearningMastery.com. Retrieved November 24, 2022, from <https://machinelearningmastery.com/what-is-deep-learning/>
3. Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., &amp; Farhan, L. (2021, March 31). Review of Deep Learning: Concepts, CNN Architectures, challenges, applications, future directions - Journal of Big Data. SpringerOpen. Retrieved November 24, 2022, from <https://journalofbigdata.springeropen.com/articles/10.1186/s40537-021-00444-8>
4. Yegulalp, S. (2022, June 3). What is tensorflow? The Machine Learning Library explained. InfoWorld. Retrieved November 25, 2022, from <https://www.infoworld.com/article/3278008/what-is-tensorflow-the-machine-learning-library-explained.html>
5. Brownlee, J. (2020, August 14). What is the difference between test and validation datasets? MachineLearningMastery.com. Retrieved November 25, 2022, from <https://machinelearningmastery.com/difference-test-validation-datasets/>
6. Sonal. (2021, September 20). Importance of exploratory data analysis before ML Modelling. Eduonix Blog. Retrieved November 25, 2022, from <https://blog.eduonix.com/bigdata-and-hadoop/importance-exploratory-data-analysis-ml-modelling/>
7. Verma, J. (2022, August 3). Mnist dataset in Python - Basic Importing and plotting. DigitalOcean. Retrieved November 25, 2022, from <https://www.digitalocean.com/community/tutorials/mnist-dataset-in-python>
8. Yegulalp, S. (2022, June 3). What is tensorflow? The Machine Learning Library explained. InfoWorld. Retrieved November 25, 2022, from <https://www.infoworld.com/article/3278008/what-is-tensorflow-the-machine-learning-library-explained.html>
9. Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., & Farhan, L. (2021, March 31). *Review of Deep Learning: Concepts, CNN Architectures, challenges, applications, future directions - Journal of Big Data*. SpringerOpen. Retrieved November 24, 2022, from https://journalofbigdata.springeropen.com/articles/10.1186/s40537-021-00444-8
10. khandelwal, S. (2020, November 26). *Cutmix: A new strategy for data augmentation*. Medium. Retrieved November 25, 2022, from https://sarthakforwet.medium.com/cutmix-a-new-strategy-for-data-augmentation-bbc1c3d29aab#27f9
11. Nair, A. (2022, April 4). *Baseline models: Your guide for model building*. Medium. Retrieved November 25, 2022, from <https://towardsdatascience.com/baseline-models-your-guide-for-model-building-1ec3aa244b8d#:~:text=A%20baseline%20model%20is%20essentially,a%20necessity%20for%20many%20reasons>.
12. Learning, G. (2021, September 23). *Everything you need to know about VGG16*. Medium. Retrieved November 25, 2022, from https://medium.com/@mygreatlearning/everything-you-need-to-know-about-vgg16-7315defb5918
13. Reed, N. (2020, December 3). *ELI5: What is image classification in deep learning?* ThinkAutomation. Retrieved November 24, 2022, from <https://www.thinkautomation.com/eli5/eli5-what-is-image-classification-in-deep-learning/>
14. Pröve, P.-L. (2017, October 18). Squeeze-and-excitation networks. Medium. Retrieved November 25, 2022, from <https://towardsdatascience.com/squeeze-and-excitation-networks-9ef5e71eacd7>
15. Nair, A. (2021, October 25). *Improve your hyperparameter tuning experience with the random search*. Medium. Retrieved November 25, 2022, from https://towardsdatascience.com/improve-your-hyperparameter-tuning-experience-with-the-random-search-2c05d789175f