OAKLAND UNIVERSITY™

School of Engineering and Computer Science

Project Report

# A Study on Gradient Descent Algorithms for Image Classification

**By**

Sai Krishna Karthikeya Challa

**Course**

Artificial Intelligence

**Professor**

Dr. Tianle Ma

# Introduction

**Neural Network:**

Neural network is a network of neurons also called as nodes for solving artificial intelligence problems. A neural network is a layered structure with input, hidden and output layers as depicted in Figure 1 below.

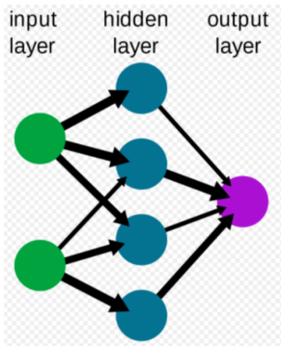


Figure 1: A simple neural network

As the complexity of the network increases, the number of hidden layers increases. Each of the nodes is connected to one or more nodes through weights and biases which are the model parameters. The magnitude of the weights and biases determines the importance of that particular node in determining the model.

**Learning:**

Neural networks are used for supervised learning in applications like image detection, image classification etc. So, they undergo a learning process also termed as training by processing examples with input and results. Each learning iteration determines the accuracy of the model which enables the calculation of the error which is the difference between prediction and actual value.

The aim of the neural network is to reduce the error and minimize the loss. This improves accuracy of the model. This is possible by updating the weights and biases which are the internal learning parameters. These parameters are updated based on some rules called Gradient Descent.

**Gradient Descent:**

It is an iterative mathematical operation for finding the minimum values of a function. So gradient descent when applied on the loss function enables us to achieve the point of lowest loss and improves the accuracy. So, in the learning process, the weights and biases are updated multiple times to achieve the minimum value for the loss function. Figure 2 below represents the gradient descent.

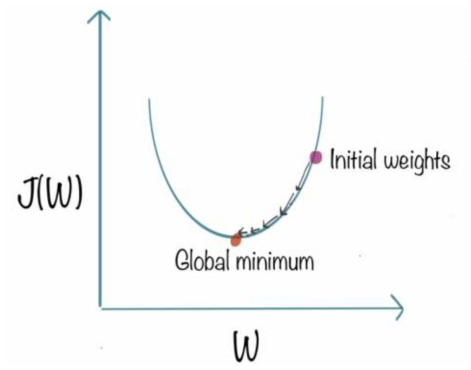


Figure 2: A depiction of gradient descent

**Gradient Descent Algorithms:**

Recent times have seen a significant research towards optimizing the update rules for improving the accuracy of the models. Many gradient descent algorithms also called optimization algorithms have been proposed and are in usage.

Some important ones:

- SGD – Stochastic Gradient Descent
- Momentum
- ADAM
- ADAMAX
- RMSPROP
- ADAGRAD
- ADADELTA

Each of these optimization algorithms has its own update rule for weights and bias updates which determines the loss and accuracy of the model.

# Project aim, datasets and methodology

**Project Aim:**

With large sets of data in various genres, with multiple models and advancement in complexity of the neural networks and with many optimization algorithms, it becomes difficult to choose one particular optimization algorithm. It is computationally intensive and challenging to re-run the models with various algorithms which makes this study an interesting and fetching one.

With focus on image classification example, this project aims at studying the various gradient descent algorithms and their performance for improving the accuracy of the models for image classification.

**Datasets:**

The datasets used for this project are obtained from Tensorflow Datasets.

1) Cassava Leaf ( https://www.tensorflow.org/datasets/catalog/cassava )
   Cassava consists of leaf images for the cassava plant depicting healthy and four (4) disease conditions; Cassava Mosaic Disease (CMD), Cassava Bacterial Blight (CBB), Cassava Greem Mite (CGM) and Cassava Brown Streak Disease (CBSD). Dataset consists of a total of 9430 labelled images. Total number of classes are 5.

2) Oxford Flowers 102 ( https://www.tensorflow.org/datasets/catalog/oxford_flowers102 )
   The dataset is divided into a training set, a validation set and a test set. The training set and validation set each consist of 10 images per class (totaling 1020 images each). The test set consists of the remaining 6149 images (minimum 20 per class). Total number of images in this dataset are 8189 and number of classes is 102.

For uniformity of the data analysis each of these datasets is divided into 70% training and 15% each for validation and testing sets.

Training sets help train the model, while the validation set enables performance improvement and determines the true nature of the model developed.

Bettering the model based on validation accuracy and loss is an essential part of training process which helps in obtaining better accuracy on the testing set.

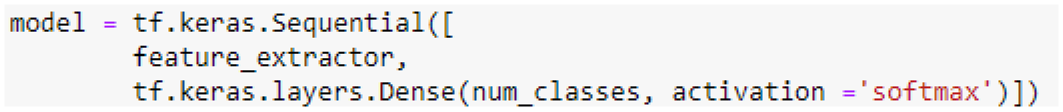
**Transfer Learning Models:**

The process where already trained model termed pre-trained model is used without modifying its learning parameters and by adding a final output layer for obtaining the final model is known as transfer learning.

It is efficient to adapt and already proven model for image classification problems. So, 3 such famous models are used form Tensorflow HUB which are:

- Mobilenet ( https://tfhub.dev/google/tf2-preview/mobilenet_v2/feature_vector/4 )
- Resent50 ( https://tfhub.dev/tensorflow/resnet_50/feature_vector/1 )
- Inception ( https://tfhub.dev/google/imagenet/inception_v3/feature_vector/4 )

At the end of each model a dense layer with number of classes in the datasets as output is added for the output layer with a softmax activation function shown in Figure 3 below.



```
model = tf.keras.Sequential([
        feature_extractor,
        tf.keras.layers.Dense(num_classes, activation ='softmax')])
```

Figure 3: Pre-Trained model with output dense layer

**Optimization Algorithms and parameters:**

For model compilation the following optimizers are used:

- SGD
- ADAGRAD
- ADAM
- RMSPROP

The following parameters are used:

Number of epochs: 25

Batch Size: 32

Learning rate: 0.001 with exponential decay

Loss Function: Sparse Categorical Entropy

Metric: Accuracy

# Experiments and Results

The transfer learning models have been applied to the mentioned datasets with respective parameters and optimization algorithms.

The training accuracy and validation accuracy plots with respect to number of epochs have been plotted to understand the pros and cons of each of the optimization algorithms. Also, the accuracy metric for test set is calculated to conclude which gradient descent algorithm is best for the image classification application.

**Pros and Cons of SGD – Stochastic Gradient Descent:**

SGD updates the parameters for each observation and hence it is faster. But it is only a stochastic approximation which can make the test accuracy poor. Also a fixed learning rate makes it faster and a bit inaccurate. Due this approximation, the algorithm may get stuck at a local minimum causing lot a greater number of epochs increasing the learning time and computing capability. But for lower number of epochs the validation accuracy will be poor as in Figure 4 below.

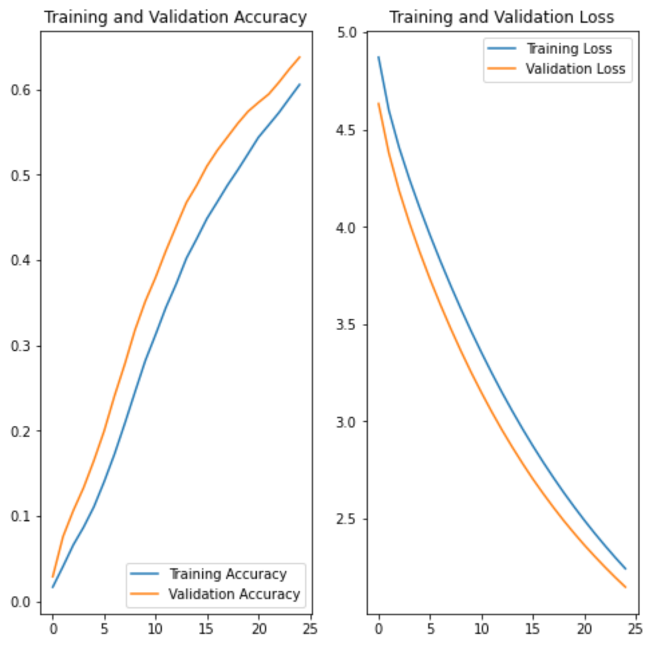


Figure 4: Training and Validation Accuracy and Loss plots for SGD

**Pros and Cons of ADAGRAD:**

ADAGARD uses the adaptive learning rate for updating the parameters. This gives ADAGRAD an edge over SGD where the algorithm can converge at a faster rate and improve the accuracy.

ADAGRAD uses a different learning for each parameter. It is a running sum of squares of gradients for each parameter which makes learning rate high or low for low or high accumulated sums respectively. Therefore, the learning rate decays aggressively making the algorithm slow and ends up with poorer accuracy while converging slowly as depicted in Figure 5 below.

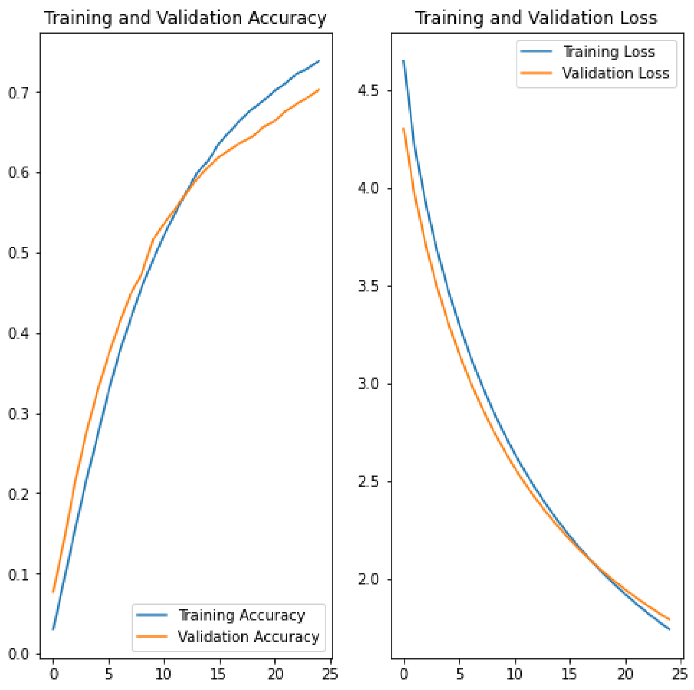


Figure 5: Training and Validation Accuracy and Loss plots for ADAGRAD

**Pros and Cons of RMSProp:**

RMSProp overcomes the issue of decaying learning rate as it accumulates only a fraction of squares of gradients. This allows the algorithm to keep learning and ultimately achieve higher accuracy and tries to converge.

RMSProp adopts adaptive learning rate methodology which makes it faster too.

But the issue with RMSProp is that, the algorithm traverses through each and every local minimum and hence number of oscillations increase.

Figure 6 below shows the oscillatory nature of the RMSProp but improving the accuracy.

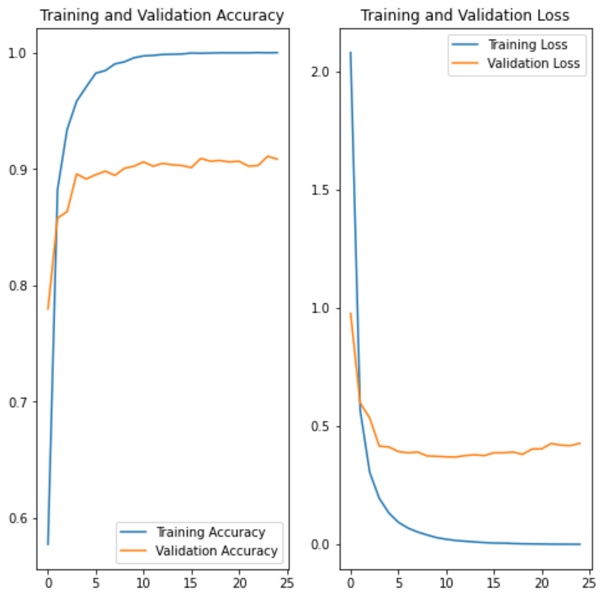


Figure 6: Training and Validation Accuracy and Loss plots for RMSProp

**Pros and Cons of ADAM:**

Along with adaptive learning rate, ADAM adopts the adaptive momentum which makes it a best combination from SGD Momentum and RMSProp. It will be stochastic approximation making it smoother and faster than RMSProp as shown in Figure 7 below. The algorithm converges faster and keeps learning at an adaptive rate with some momentum and hence doesn't get stuck at any local minimums.

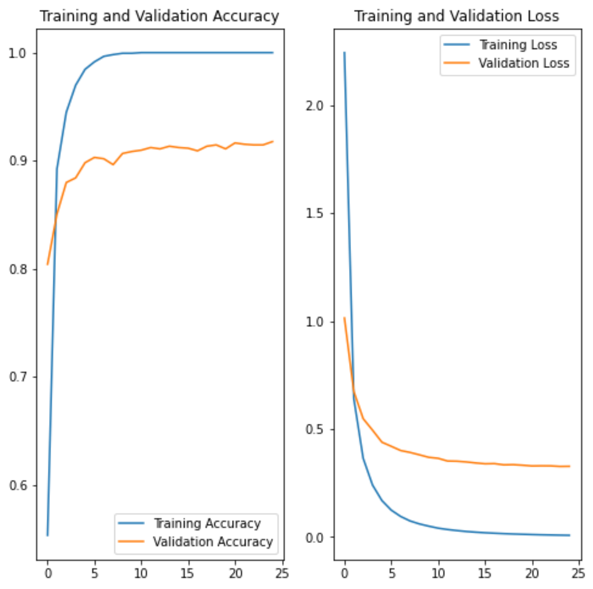
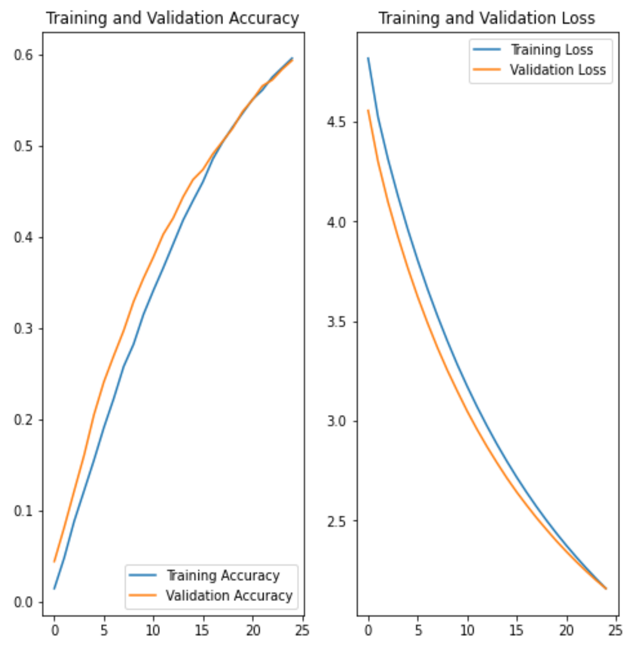
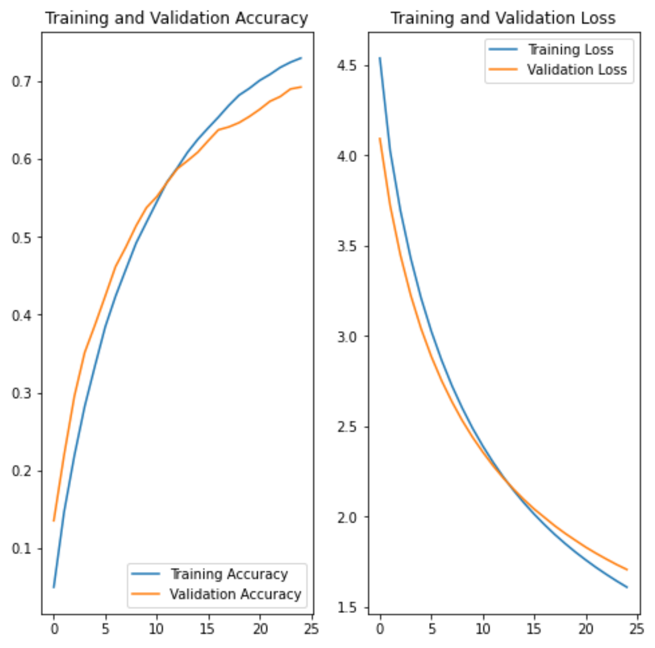
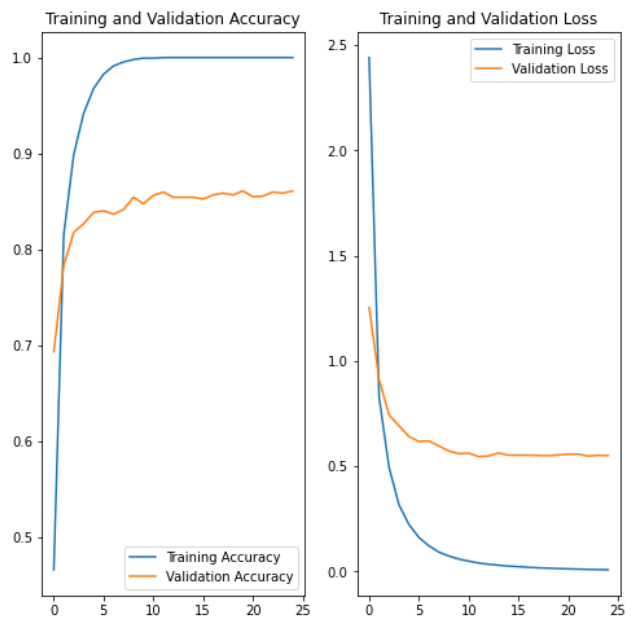
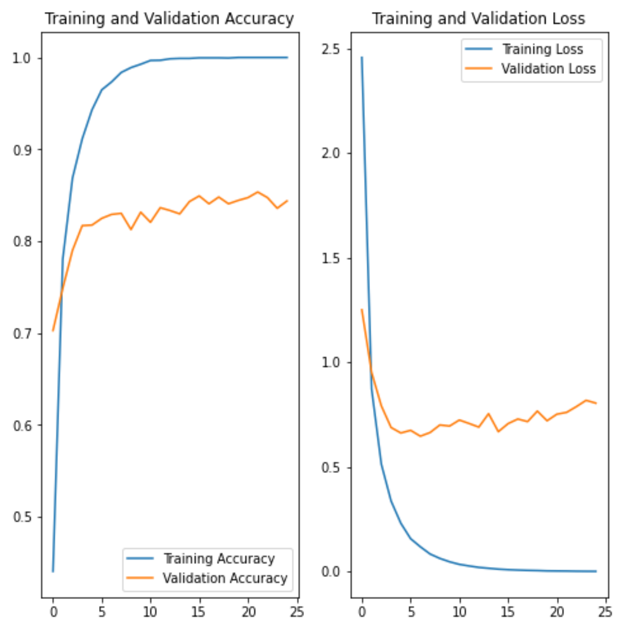
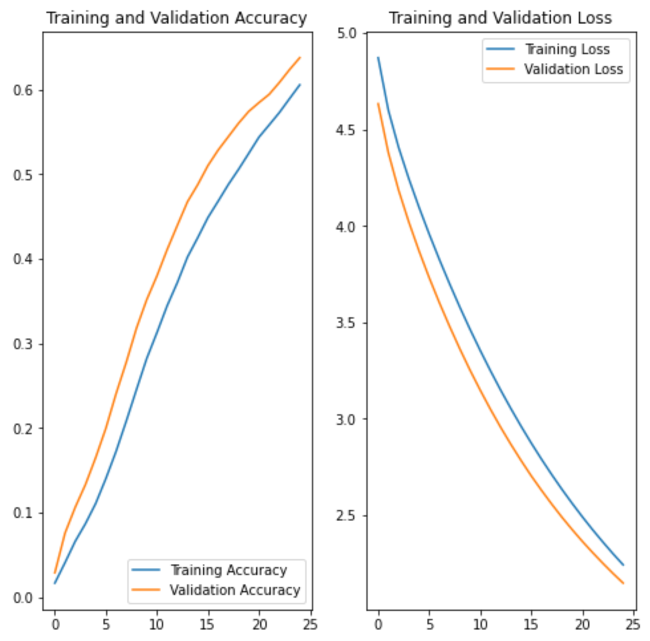
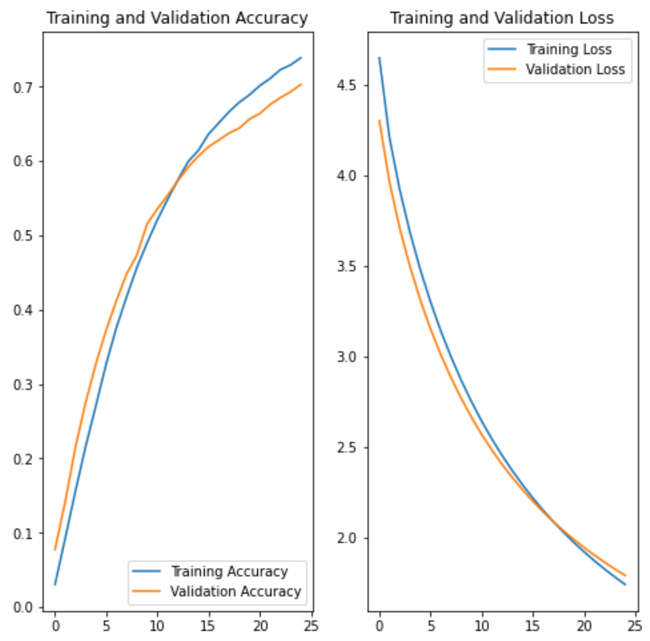
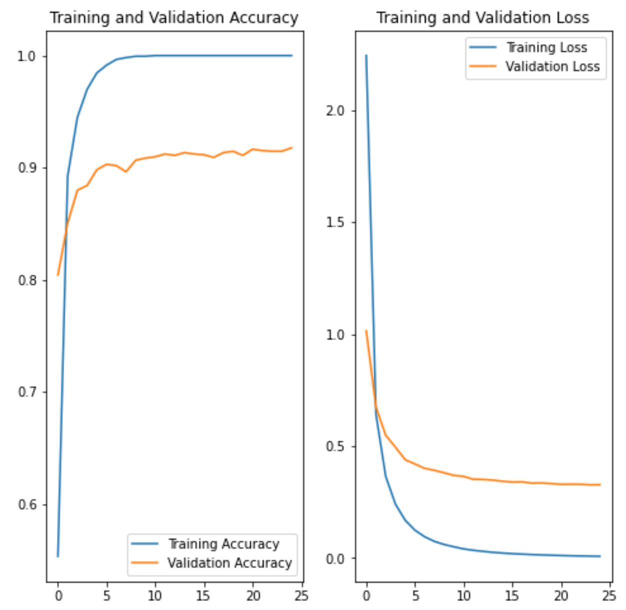
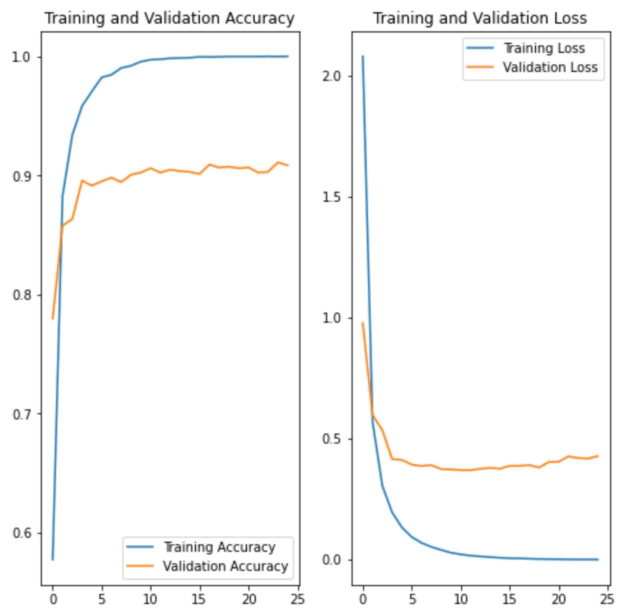


Figure 7: Training and Validation Accuracy and Loss plots for ADAM

**Plots for Inception model on Oxford Flowers and optimizers:**

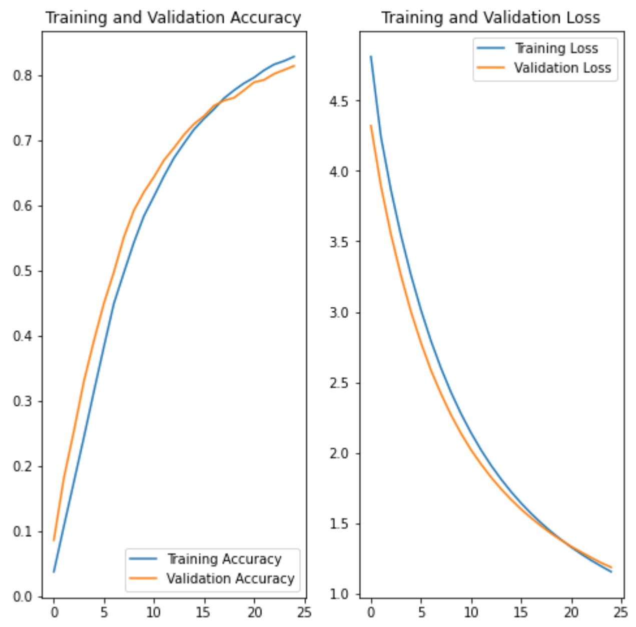
| SGD | ADAGRAD |
|---|---|
|  |  |
| **RMSProp** | **ADAM** |
|  |  |

**Plots for Mobilenet model on Oxford Flowers and optimizers:**

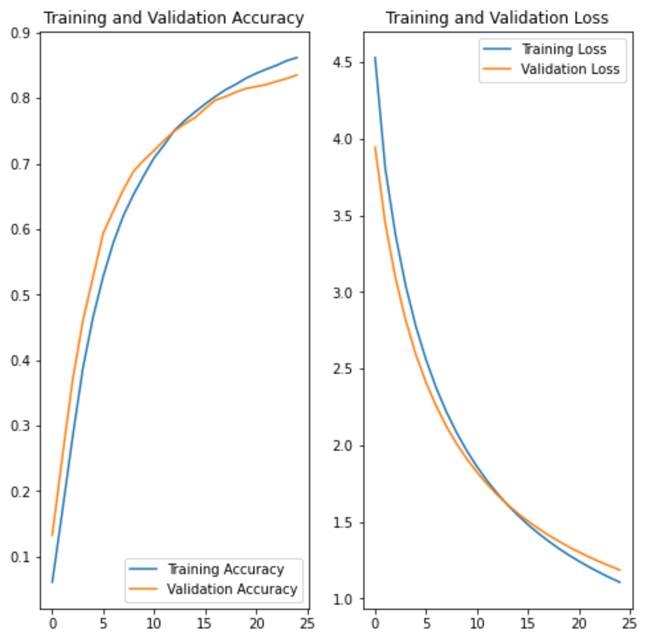
| **SGD** | **ADAGRAD** |
|---|---|
|  |  |
| **RMSProp** | **ADAM** |
|  |  |

**Plots for Resnet50 model on Oxford Flowers and optimizers:**

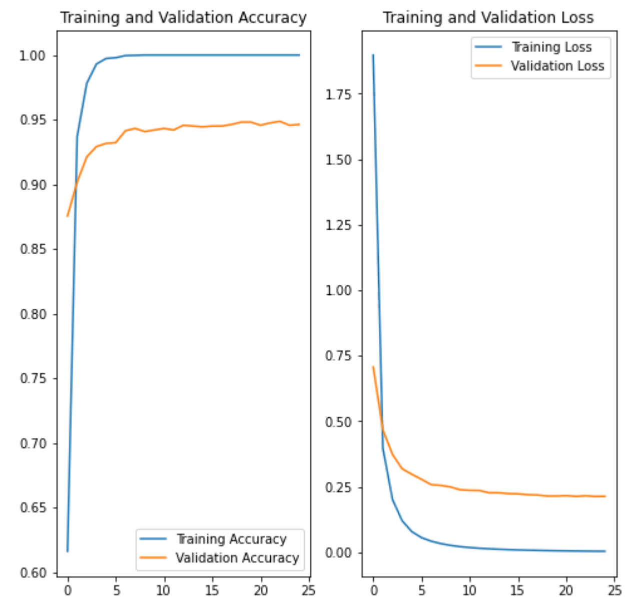
**SGD**



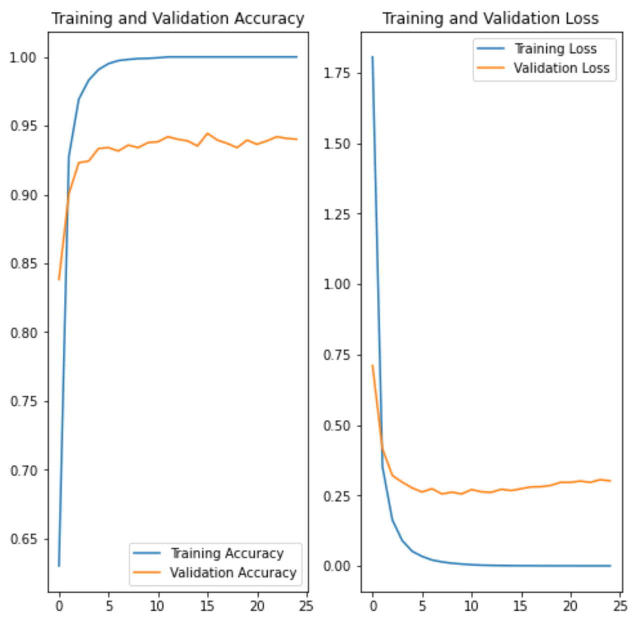
**ADAGRAD**



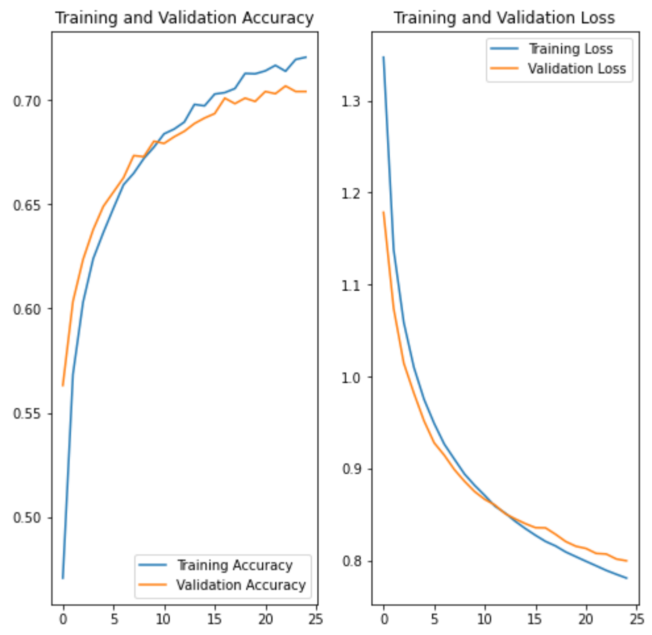
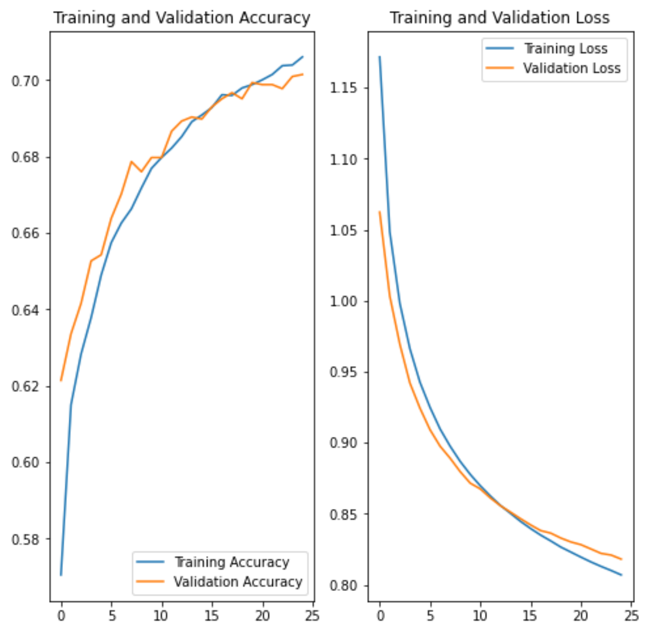
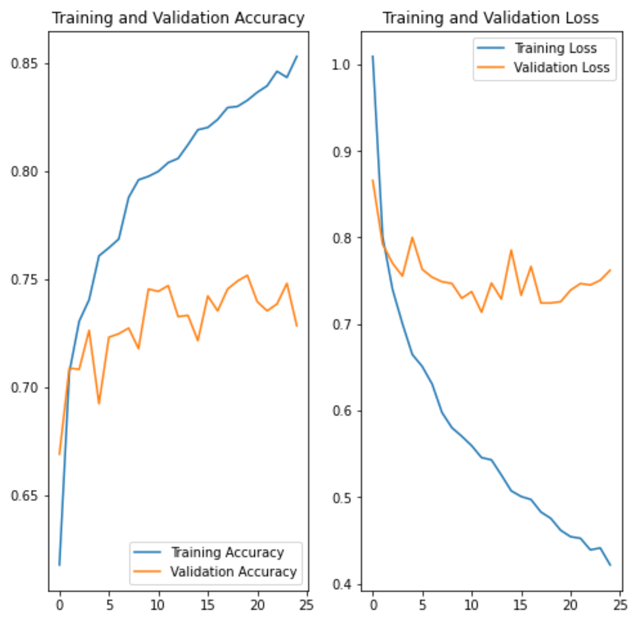
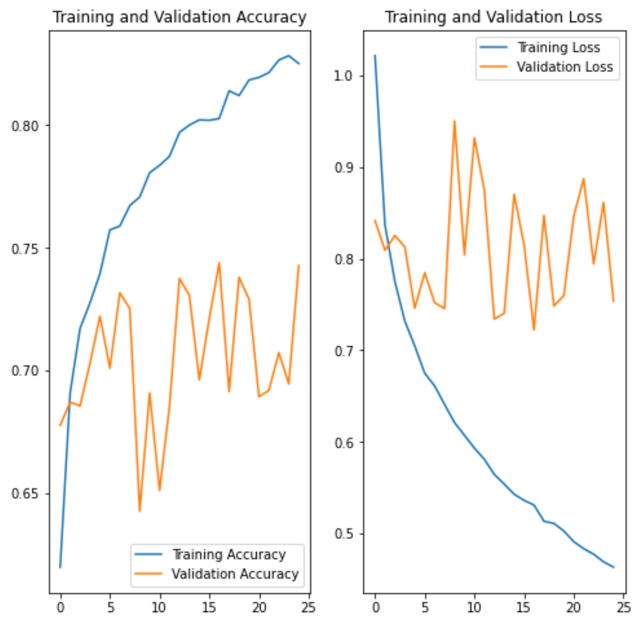
**RMSProp**



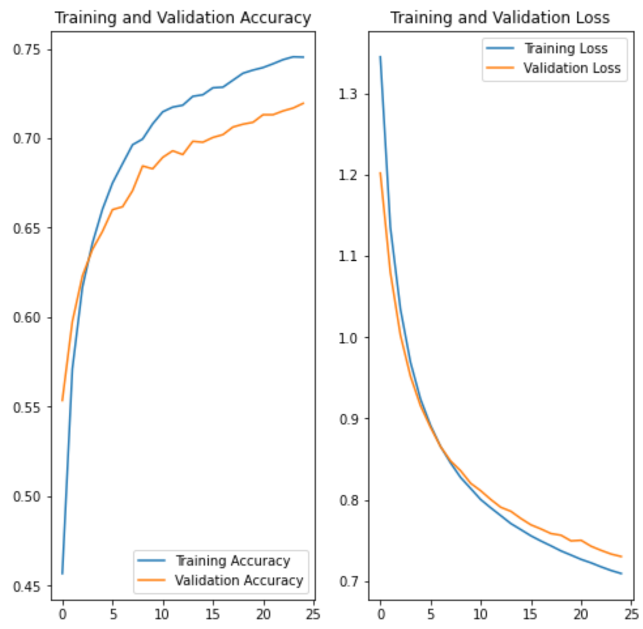
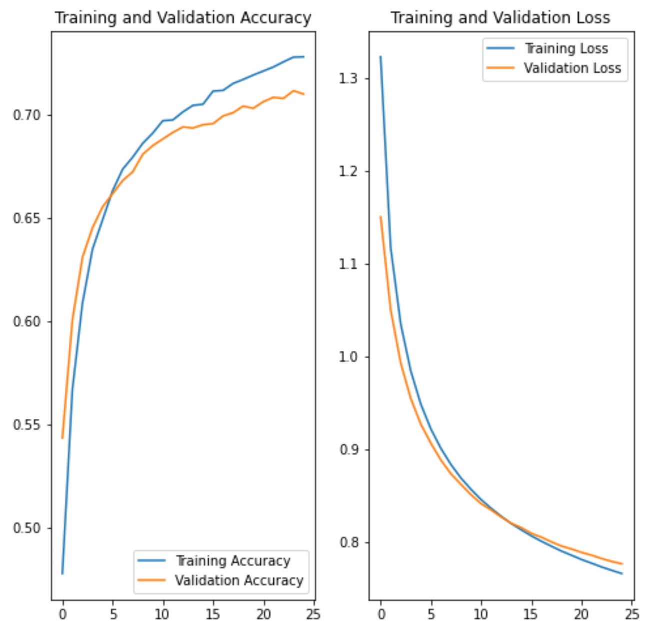
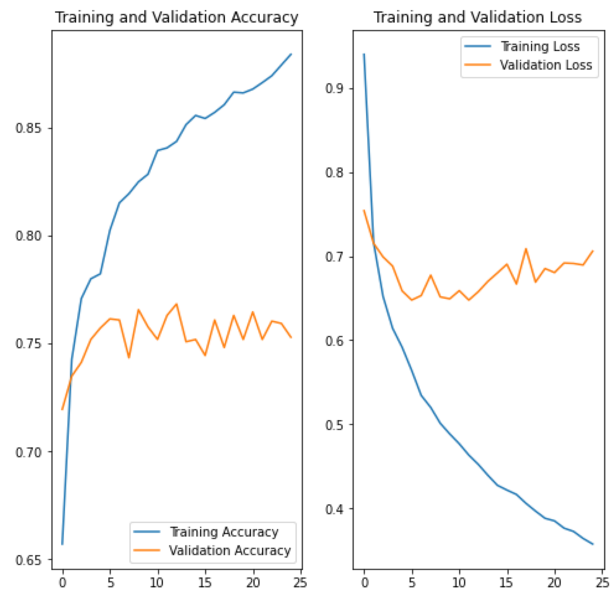
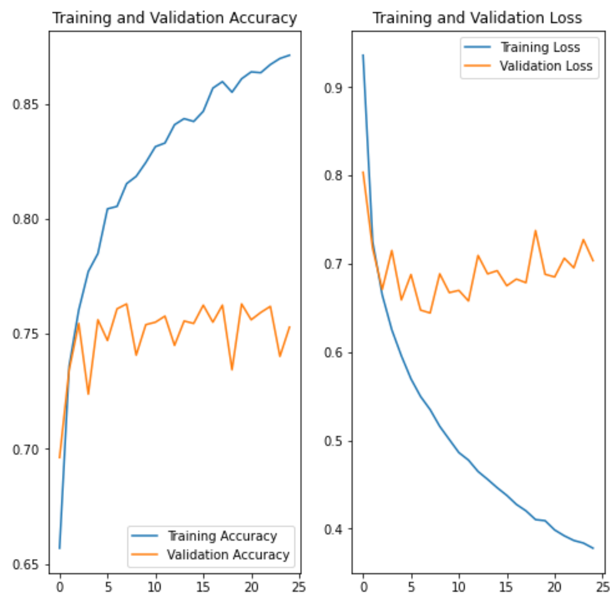
**ADAM**



**Plots for Inception model on Cassava Leaf and optimizers:**

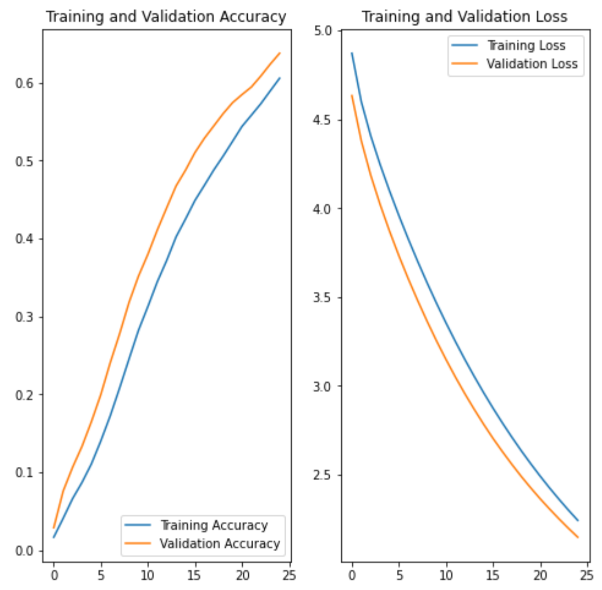
| SGD | ADAGRAD |
|---|---|
|  |  |
| **RMSProp** | **ADAM** |
|  |  |

**Plots for Mobilenet model on Cassava Leaf and optimizers:**

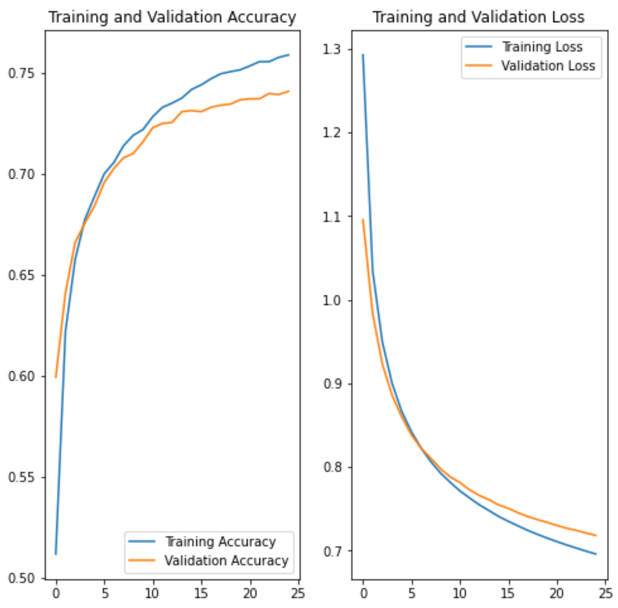
| SGD | ADAGRAD |
|---|---|
|  |  |
| **RMSProp** | **ADAM** |
|  |  |

**Plots for Resnet50 model on Cassava Leaf and optimizers:**

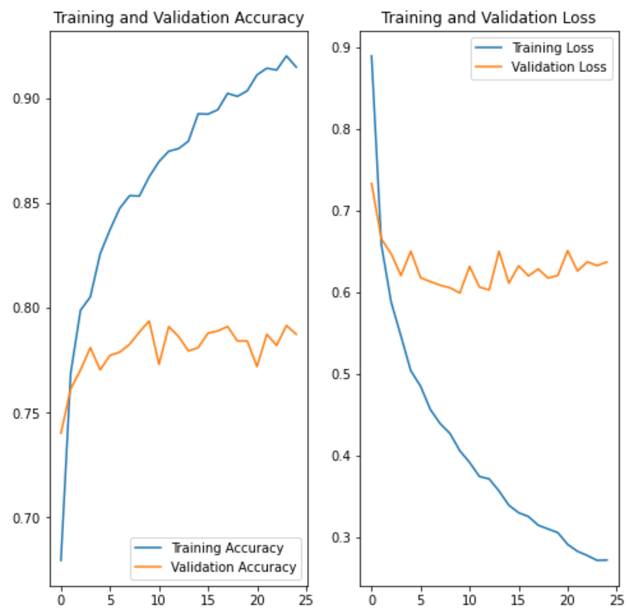
**SGD**



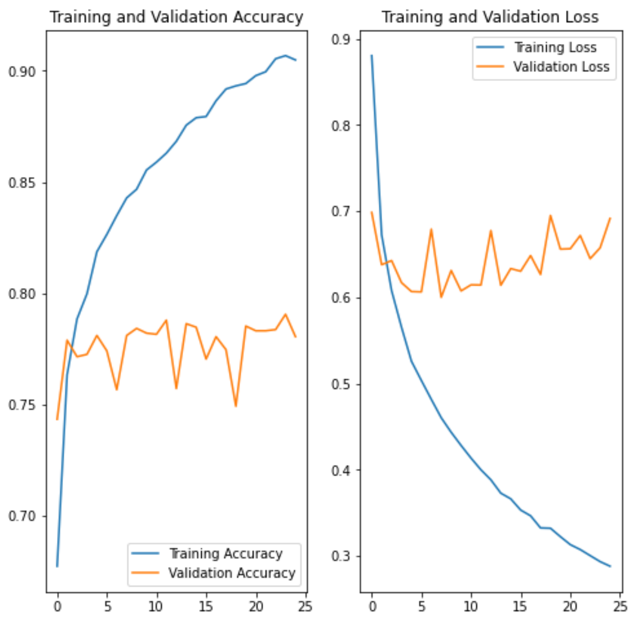
**ADAGRAD**



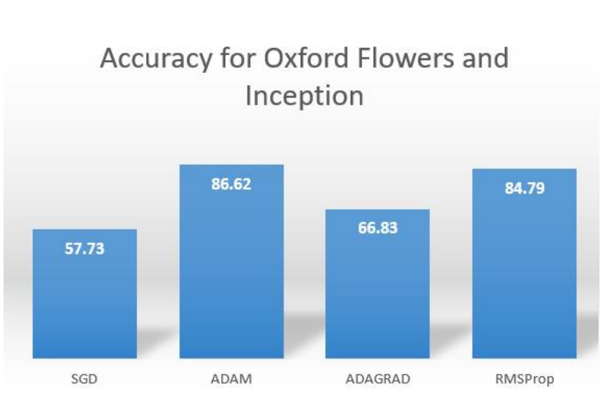
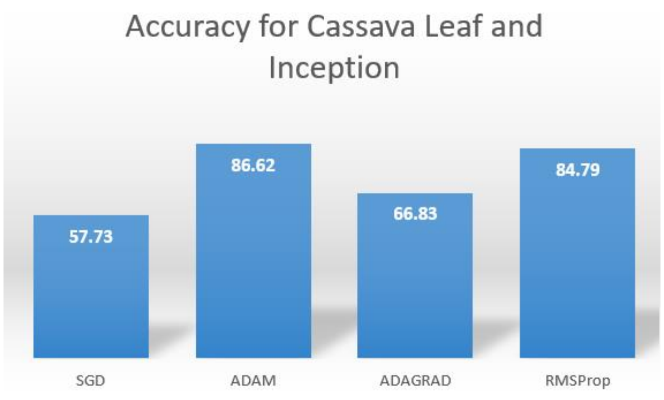
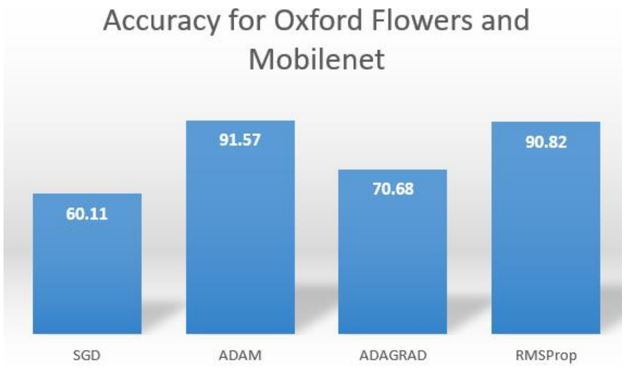
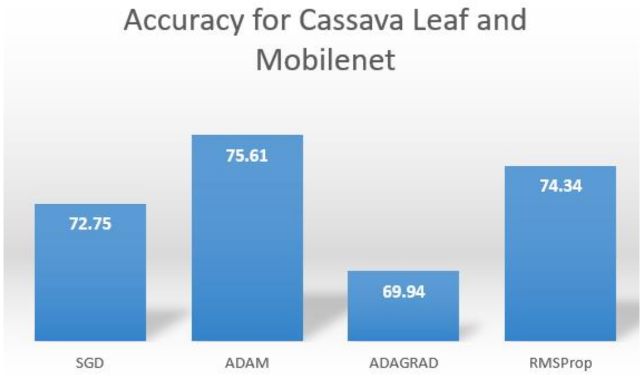
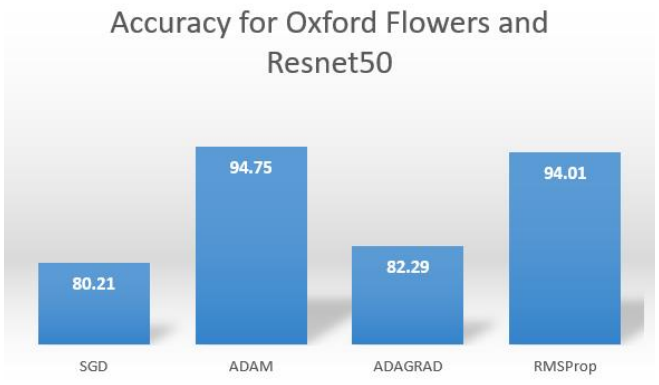
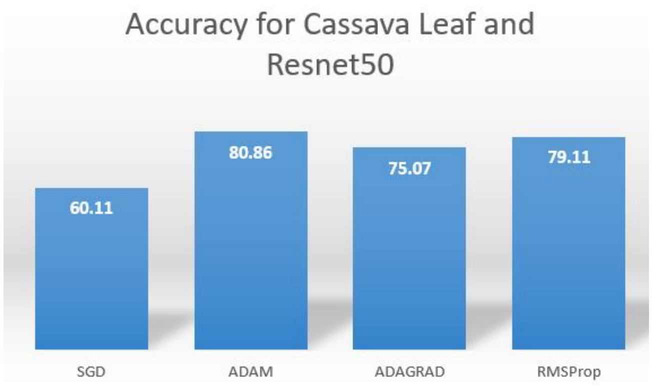
**RMSProp**



**ADAM**



**Plots for accuracy on test sets for all models and optimizers:**

|  |  |
|---|---|
|  |  |
|  |  |
|  |  |

# Conclusion

The image classification application has been performed on Cassava leaf and Oxford flowers datasets from Tensorflow Datasets using mobilenet, inception and resnet50 pre-trained models from Tensorflow HUB. Various optimizers namely SGD, ADAM, ADAGRAD and RMSProp have been used for model compilation. All the models are run with same parameters like number of epochs, batch size and learning rate.

In this project two questions have been answered:

1) What are the pros and cons of various algorithms for image classification application?

➔ Every optimizer has its own pros and cons. SGD is faster while inaccurate, ADAGRAD is fast but has decaying learning rate, RMSProp is fast and adapts to learning rate but has many oscillations while ADAM is faster and smoother. These behaviors can be observed from the loss and accuracy plots for various optimizers.

2) Which GD algorithm is the best for compiling the model?

➔ From the accuracy on test set plots, it can be observed that the ADAM algorithm is the best for image classification. It is model agnostic and is always better performing with higher accuracy.

The below table concludes the accuracy observations on test set.

| Model | Optimizer | Accuracy for Oxford Flowers | Accuracy for Cassava Leaf |
|---|---|---|---|
| Inception | SGD | 57.73% | 57.73% |
| Inception | ADAGRAD | 66.83% | 66.83% |
| Inception | RMSProp | 84.79% | 84.79% |
| Inception | ADAM | 86.62% | 86.62% |
| Mobilenet | SGD | 60.11% | 72.75% |
| Mobilenet | ADAGRAD | 70.68% | 69.94% |
| Mobilenet | RMSProp | 90.82% | 74.34% |
| Mobilenet | ADAM | 91.57% | 75.61% |
| Resnet50 | SGD | 80.21% | 60.11% |
| Resnet50 | ADAGRAD | 82.29% | 75.07% |
| Resnet50 | RMSProp | 94.01% | 79.11% |
| Resnet50 | ADAM | 94.75% | 80.86% |

# Related Work and References

[1] S. Vani and T. V. M. Rao, "An Experimental Approach towards the Performance Assessment of Various Optimizers on Convolutional Neural Network," *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*, Tirunelveli, India, 2019, pp. 331-336, doi: 10.1109/ICOEI.2019.8862686.

[2] Z. Zhang, "Improved Adam Optimizer for Deep Neural Networks," *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*, Banff, AB, Canada, 2018, pp. 1-2, doi: 10.1109/IWQoS.2018.8624183.

[3] Keskar, N., & Socher, R. (2017, December 20). Improving Generalization Performance by switching from Adam to SGD. Retrieved December 09, 2020, from https://arxiv.org/abs/1712.07628

[4] Luhaniwal, V. (2020, October 24). Why Gradient descent isn't enough: A comprehensive introduction to optimization algorithms in... Retrieved December 09, 2020, from https://towardsdatascience.com/why-gradient-descent-isnt-enough-a-comprehensive-introduction-to-optimization-algorithms-in-59670fd5c096

[5] Neural network. (2020, November 30). Retrieved December 09, 2020, from https://en.wikipedia.org/wiki/Neural_network

[6] Gradient descent. (2020, December 03). Retrieved December 09, 2020, from https://en.wikipedia.org/wiki/Gradient_descent

[7] Dabbura, I. (2019, September 03). Gradient Descent Algorithm and Its Variants. Retrieved December 09, 2020, from https://towardsdatascience.com/gradient-descent-algorithm-and-its-variants-10f652806a3