NOVEMBER 2021

Deep Learning Based Seed Classification and Quality Detection

A PROJECT REPORT

Submitted by

BL.EN.U4CSE18069	M.Saatwika
BL.EN.U4CSE18112	Sanka Devadutt
BL.EN.U4CSE18133	Varanasi Madhulika

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

AMRITA SCHOOL OF ENGINEERING, BANGALORE

AMRITA VISHWA VIDYAPEETHAM

BANGALORE 560 035

June-2021

NOVEMBER 2021

AMRITA VISHWA VIDYAPEETHAM

AMRITA SCHOOL OF ENGINEERING, BANGALORE, 560035

BONAFIDE CERTIFICATE

This is to certify that the project report entitled **“Deep learning based seed Classification and quality detection”** submitted by -

BL.EN.U4CSE18069	M. Saatwika
BL.EN.U4CSE18112	Sanka Devadutt
BL.EN.U4CSE18133	Varanasi Madhulika

In partial fulfillment of the requirements for the award of the **Degree Bachelor of Technology** in “**COMPUTER SCIENCE AND ENGINEERING** “is a bonafide record of the work carried out under my guidance and supervision at Amrita School of Engineering, Bangalore.

[illegible] 01/12/2021

Dr. B. Uma Maheswari	Dr.Sriram Devanathan
Assistant Professor(SG)	Principal & Chairperson,
Computer Science and Engineering	Computer Science and Engineering

This project report was evaluated by us on .29/11/2021

EXAMINER1 EXAMINER2

ACKNOWLEDGEMENT

The satisfaction that accompanies successful completion of any task would be incomplete without mention of people whom made it possible, and whose constant encouragement and guidance have been source of inspiration throughout the course of this project work.

We offer our sincere pranams at the lotus feet of**”AMMA”,MATA AMRITANANDAMAYI DEVI** who showered her blessing upon us throughout the course of this project work.

We owe our gratitude to **Br.Viswamrita Chaitanya Swamiji**, Director, Amrita School of Engineering, Bangalore.

We thank **Dr. Sriram Devanathan**, Principal, Amrita School of Engineering, Bangalore for his support and inspiration.

It is a great pleasure to express our gratitude and indebtedness to our project guide **Dr. B. Uma Maheswari**., Assistant Professor(SG), Department of Computer Science and Engineering, Amrita School of Engineering, Bangalore for her valuable guidance, encouragement, moral support and affection throughout the project work.

We express our Heartfelt thanks to **Dr. Sriram Devanathan,** Chairperson, Department of Computer Science and Engineering, Amrita School of Engineering, Bangalore for his encouragement and support during the course of work.

We would like to thank express our gratitude to project panel members for their suggestions, encouragement and moral support during the process of project work.Finally, we are forever grateful to our parents, who have loved, supported and encouraged us in all our endeavors.

NOVEMBER 2021

ABSTRACT

Seeds contain all of the necessary ingredients to grow into complex plants. They're incredibly nutritious as a result of this. Seeds contain a lot of fibre. They're also high in healthful fats, as well as vitamins, minerals, and antioxidants. Seeds can help lower blood glucose, cholesterol, and blood pressure levels when ingested as part of a healthy diet. Seeds are classified based on their colour, shape, and texture in general. This method necessitates the extraction of a large amount of feature data over and over again, which is inefficient in detection. Deep learning has done admirably in the field of image identification in recent years. For seed classification, we used convolutional neural networks (CNNs). We primarily concentrated on detecting the quality and type of seeds for moongdal, pigeon pea, and groundnuts in our project.

NOVEMBER 2021

TABLE OF CONTENTS

	Page no
ACKNOWLEDGEMENT	iii
ABSTRACT	iv
LIST OF FIGURES	v
CHAPTER 1- INTRODUCTION	01
1.1 INTRODUCTION TO CNN	02
1.2 OBJECTIVE	02
CHAPTER 2 – LITERATURE REVIEW	03
2.1 LITERATURE SURVEY	03
2.2 GAP IDENTIFIED	05
2.3 PROBLEM STATEMENT	06
CHAPTER 3 – REQUIREMENTS	07
3.1 SOFTWARE AND HARDWARE REQUIREMENTS	07
CHAPTER 4 – DESIGN	08
4.1 DATA PREPROCESSING	08
4.2 MODEL GENERATION	09
4.3 MODEL VALIDATION	09
4.4 MODEL TESTING	09
CHAPTER 5 – IMPLEMENTATION	10
5.1 BUILDING DATASET	10
5.2 DATA PREPROCESSING	11
5.3 TRAIN AND TEST SPLIT	12
5.4 MODEL CREATION	13
5.5 IMAGE AUGMENTATION	14
5.6 MODEL TRAINING	14
5.7 MODEL TESTING	15
CHAPTER 6 – SYSTEM TESTING AND TESTCASES	17

NOVEMBER 2021

CHAPTER 7 – RESULTS AND DISCUSSIONS	27
CHAPTER 8 – CONCLUSION AND FUTURE ENHANCEMENT	29
REFERENCES	30

LIST OF FIGURES

Page no

Fig 1.1: CNN model representation in general	02
Fig 4.1: Process flow diagram	08
Fig 5.1: Seed types and Image count	09
Fig 5.2: Dataset distribution	09
Fig 5.3: Images of three different seeds good and bad quality	10
Fig 5.4: Data Preprocessing	11
Fig 5.5: Reshaping Numpy arrays	11
Fig 5.6: Train and test split	11
Fig 5.7: Importing pre-trained model	12
Fig 5.8: Freezing layers in the model	12
Fig 5.9: Modification of the model	12
Fig 5.10: Model compilation	12
Fig 5.11: Image Augmentation	13
Fig 5.12: Model Training	13
Fig 5.13: Training/Validation Accuracy	14
Fig 5.14: Training/Validation Loss	15
Fig 5.15: Model Testing	15
Fig 5.16: Classification report	16
Fig 5.17: Confusion Matrix of testing	16
Fig 7.1: Model prediction on unseen data	27
Fig 7.2: Classification report on new data	27
Fig 7.3: Confusion Matrix of unseen data	28

CHAPTER 1 – INTRODUCTION

The agriculture industry employs the majority of the world's population. Agriculture is crucial to the economics of many emerging and developing countries. With the growth of global population, this industry has seen various changes. The need to boost global agricultural output and sustainability is strongly tied to human population growth. Agriculture has been using technology for more than a century, and various research have been undertaken since the 1990s to increase production efficiency. AI based sophisticated industrial technologies have lately been applied in agriculture to boost productivity, ecological responsibility, food safety and sustainability. Understanding the underlying ecosystem of agriculture and the basic necessities of agriculture is required before tackling any challenge in this field. This is one of the most difficult areas of research, and the technology has huge potential for increasing the quantity and quality of agricultural goods. In the agricultural industry, advancements can be made by incorporating AI, particularly deep learning insights.

Plant growth is heavily reliant on seeds in agriculture. There are no opportunities for developing or harvesting beneficial plants without seeds. For years, the human population has been rapidly increasing. The agricultural land is shrinking day by day as a result of this population growth, resulting in a drop in food production. Crop output should be boosted to bring the consumption rate into line with the production rate. People have begun to produce and grow vegetables in their homes in this regard. However, not everyone possessed the essential skills. It can only be grown someone who has achieved seed identification. To eliminate this reliance, an automated system that can help recognize and classify different types of seeds is required. Several studies have been undertaken employing AI technologies to address various challenges linked to seeds, starting from basic object identification techniques to the identification of complex patterns and textures. Machine learning approaches have been used more often in recent research to classify seeds from various crops, fruits, and vegetables. The majority of this research focused on a single seed genus(i.e., weed seeds, cotton seeds, rice seeds, oat seeds, sunflower seeds, tomato seeds, and corn seeds) and were conducted for variety of reasons. This includes checking for life, cleanliness, and growth during the germination period.

1.1 Introduction to CNN

A few researches have used Convolutional Neural Networks (CNNs) to solve obstacles in seed identification and classification. CNNs are 4,444-layer deep learning models that include convolutional, clustering, and fully connected layers. Convolution layers are used for feature extraction, grouping layers are used for compression, and completely connected layers are used for classification. Its main job is to recognize and classify images. Fig 1.1 depicts a board representation of the CNN. With the use of precise feature extractions, CNN can make visual image analysis more efficient and accurate.

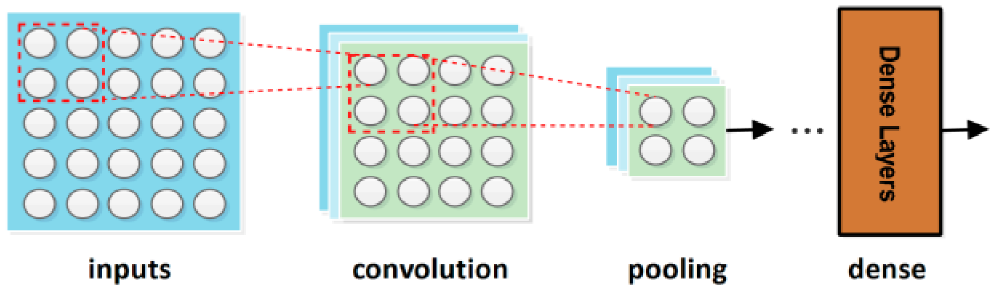


Fig 1.1 CNN model representation in general

1.2 Objective

The goal of our work is to implement a seed classifier and quality detector based on CNN to classify various species of seed based on their type and quality.

CHAPTER 2–LITERATURE REVIEW

Agriculture has profited immensely from AI because it offers answers to a number of problems that are flexible, high-performing, precise, and low-cost. For many researchers, image processing for computer vision has been a major focus. In agriculture, image processing was not frequently used. Traditional ML techniques are widely employed in AI to deal with the classification and identification of objects. Image processing involves the extraction of features, which is considered a complicated operation that aids in the identification and categorization of things. In this chapter, we look at some of the papers that describe various seed categorization approaches.

2.1 Literature Survey

Jamuna et al. [1] employed a sample of 900 cotton seeds to train the model in feature extraction using machine learning approaches (e.g., Naive Bayesian Classifier, Decision Tree Classifier, and MLP). They found MLP and decision tree classifier had similar classification precision in cotton seed classification i.e., 98.7% accuracy rate while the Naïve Bayesian classifier achieved a 94.22% accuracy rate. According to their findings, the Naïve Bayes classifier had the highest error rate, misclassifying 52 times, while in case of decision tree and MLP 11 times.

Traditional machine learning algorithms depends on user-specified characteristics for feature extraction, which might lead to the loss of critical data and make it difficult for researchers to obtain trustworthy results. Deep learning algorithms determine image quality rather than relying on the attributes of images in different levels. For instance, Rozman and Stajnko [5]conducted research on tomato seed quality in terms of vitality and germination. They proposed a computer vision system and reported on an in-depth method for image processing and extraction using a Gaussian filter, segmentation, and a region of interest (ROI), kNN, decision tree classifiers, SVM, ANN to classify 700 seeds in their study. With a precision of 95.44%, the ANN performed best in seed classification among these algorithms. NBC had an accuracy rate of 87.89% knn had a rate of 91.66%, DT had a rating of 93.66%, and SVM had an accuracy rate of 93.09%.

Agrawal and Dahiya[9] used LR, LDA, knn, SVM, decision tree classifier, Gaussian Naïve Bayes to identify different grain seeds in a comparative research. In this study, throughputs for linear and non-linear algorithms were given. The following are the precision rates for six algorithms: 91.6% for LR, 95.8% for LDA, 87.5% for knn, 88% for the decision tree classifier(CART), 88.05% for NB, and 88.71% for SVM.

A probabilistic neural network was used in another corn seed classification study (PNN). The terahertz time-domain spectroscopy (THz-TDS) was compared to the PNN algorithm for machine learning in this study, which was conducted using waveform data. Using 5-fold cross-validation, they were able to achieve a classification accuracy of 75%[6].

Furthermore, Vlasov and Fadeev[10] did research on grain seeds utilising a machine learning technique vs mechanical methods to work out the details of feature extraction using typical machine learning, which comprised image feature extraction, descriptor retrieval, and clustering. They reported on deep learning as a second method of sorting and cleaning seeds, despite their main focus being on the usual ML methodology. The deep learning approach had a classification accuracy of 95%, while standard learning had a rate of roughly 75%, according to their findings.

Another study extracted rice characteristics for classification based on physical traits like shape, colour, and texture. For comparison of classification performance, the researchers used four statistical machine learning approaches (LR, LDA, kNN, and SVM) and five previously trained models (VGG16, VGG19, Xception, InceptionV3, and InceptionResNetV2) with deep learning techniques. The SVM method had the highest precision rate (83.9%), whereas the InceptionResNetV2 model achieved the highest precision of deep learning techniques (95.15 percent).

In a maize seed cleaning investigation, the development of hybrid data sets was reported. To extract characteristics from maize seeds, the authors went through a laborious process. A histogram, texture features, and spectral characteristics made up the hybrid feature data set. Random Forest (RF), BayesNet (BN), LogitBoost (LB), and Multilayer Perceptron (MLP) classification models, as well as numerous features optimised using the cross-validation approach, were used in their research (10 times). MLP has the best classification accuracy (98.93%) among these classifiers in terms of ROI size (150 x 150).[7]

They have more detailed processes in the feature extraction technique than typical AI-based algorithms. They also require expert assistance, which reduces the efficiency of the algorithms. Deep learning (DL), unlike classic categorization learning approaches, is not confined to algorithms with a flat structure. Complex functions can be performed with tiny samples, and the most relevant functions can be extracted from a small number of training examples. The Principal Component Analysis Network of Deep Learning Technology was employed in a study by Xinshao[17] based on a sample of weed seeds (PCANet). By learning features from the dataset, this work reduced the limitations of manual feature extraction.

DL, which is largely concerned with machine learning, has progressed in providing outcomes in a variety of data analysis tasks, particularly in the field of computer vision. DL can automatically learn abstract depth functions from a depth network and use them to describe data. It's commonly employed in a variety of visual tasks. A Deep Convolutional Neural Network (DCNN) was used in conjunction with certain classic classifiers in a study on the oat seed variety, including logistic regression (LR), the RBF kernel support vector machine (RBF SVM), and the linear kernel (LINEAR SVM).

Another study used rice spectral data to test the performance of several machine learning approaches, such as K-nearest Neighbors (kNN), support vector machines (SVM), and CNN models.CNN outperformed the other two models with accuracy rates of 89.6 percent on the training set and 87 percent on the test set, according to the researchers.

A CNN investigation looked into the complexities of sunflower seed properties. Because of their texture, contaminants in sunflower seeds are difficult to detect, according to the scientists. CNN has outperformed all other methods in terms of recognising and identifying things, according to the researchers. As a result, they applied it to their study challenge. For extracting picture details, they created an eight-layer CNN model. The accuracy of the model is substantially higher than any other traditional model, according to the findings of their exhaustive testing.[14]

2.2 Gap Identified

Most of the researchers used traditional ML methods for seed classification. Only few researchers have used CNN for classification of seed varieties but mostly all of these papers were restricted to only one species.

2.3 Problem statement

We propose a new model based on CNN for classification and quality detection of seeds. In our work we intend to implement the model for 3 different species.

CHAPTER 3–REQUIREMENTS

In this chapter we discuss about the requirements that are needed for the implementation of proposed model.

3.1 Software & Hardware requirement :

Python 3.5-3.9

Tenserflow 1.15.2 – For CNN creation and modifying layers [18]

Numpy 1.18.2 – For data preprocessing [19]

Matplotlib 3.2.1 – For Visualization [20]

Windows 7 or later (64-bit)

RAM 5GB

ROM 5GB

CHAPTER 4–DESIGN

In the previous chapter we have proposed a solution for the problem and in this chapter we discuss about the design part for implementing the model.

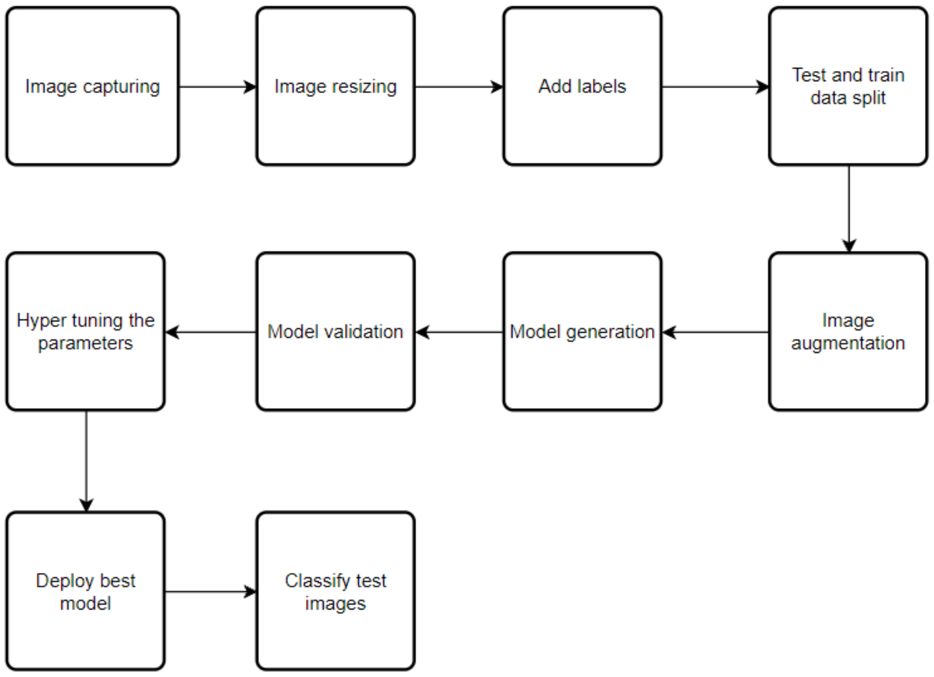


Fig 4.1: Process flow diagram

The entire procedure is divided into 3 stages : dataset creation, model testing and model training.

4.1 Data Preprocessing

The captured photos were scaled and given labels. To train the model, the dataset is separated into training and testing sets. The goal of image augmentation is to add a level of variation to the dataset in addition to extending it., which allows the model to generalize better on different datasets and also to make the model more durable when trained on fresh, slightly modified data.

4.2 Model Generation

The generation of model has 2 stages deploying base model and creating head model. A pre-trained model is deployed as base model and then we will freeze the layers of the base model. Later a head model is created with new layers on top of base model.

4.3 Model Validation

The optimizer is then used while compilation to adjust the attributes in order to minimize the losses. The model is then trained, validated, and the parameters are fine-tuned to produce the optimum model.

4.4 Model Testing

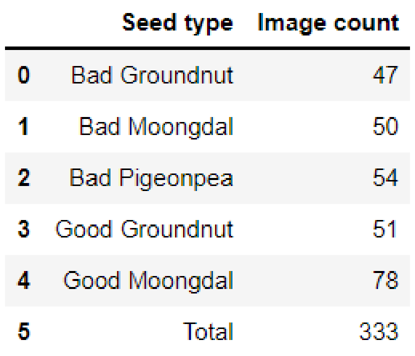
The deployed optimum model is used for classification of test-data and new data for understanding the validation of the model.

CHAPTER 5–IMPLEMENTATION

The methodology discussed in design chapter is implemented using Jupyter Notebook. The implementation of the design is as follows

5.1 Building dataset

This subdivison outlines how the dataset for this study came to be. Three different types of well-known seeds were choosen for the dataset construction in this study. Moongdal , pigeon pea ,groundnuts were the choosen seeds, as shown in Fig 5.1. Fig 5.2 depicts the frequency distribution of the dataset.



	Seed type	Image count
0	Bad Groundnut	47
1	Bad Moongdal	50
2	Bad Pigeonpea	54
3	Good Groundnut	51
4	Good Moongdal	78
5	Total	333

Fig 5.1: Seed types and Image count

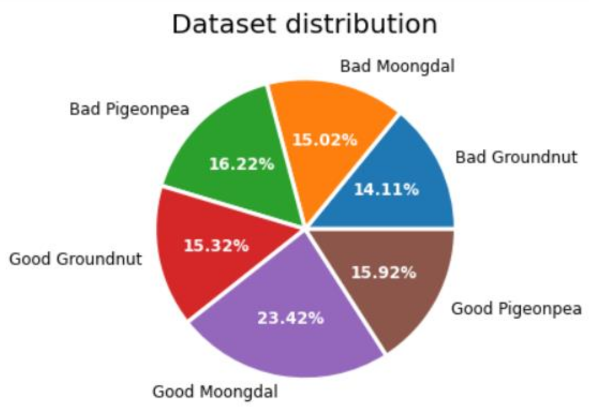


Fig 5.2: Dataset distribution

Digital Photographs of the selected seeds were taken with a smartphone(Samsung note 10 lite). The camera has a 12Mp resolution. Images of good and bad quality for each seed type can be seen in Fig 5.3.

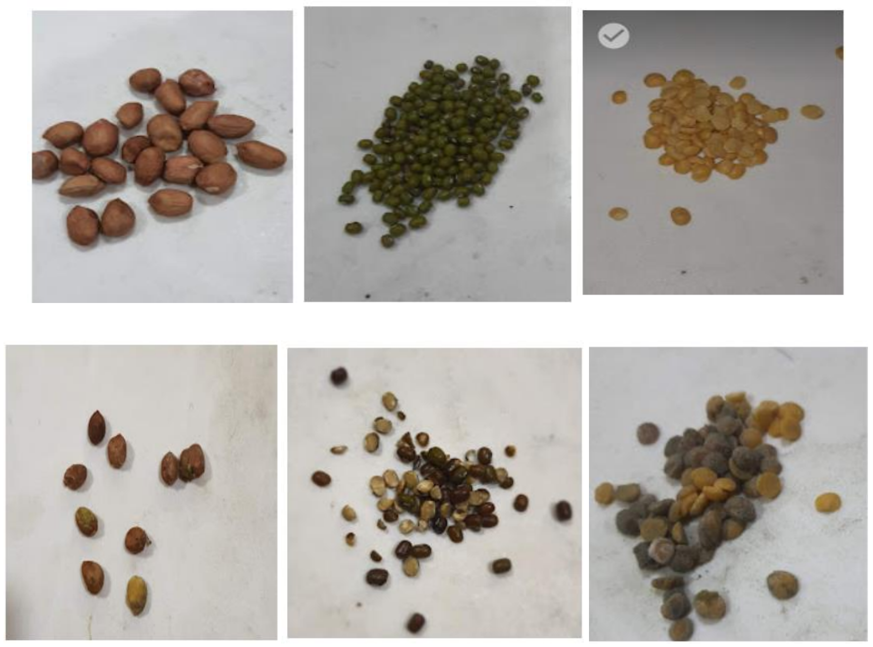
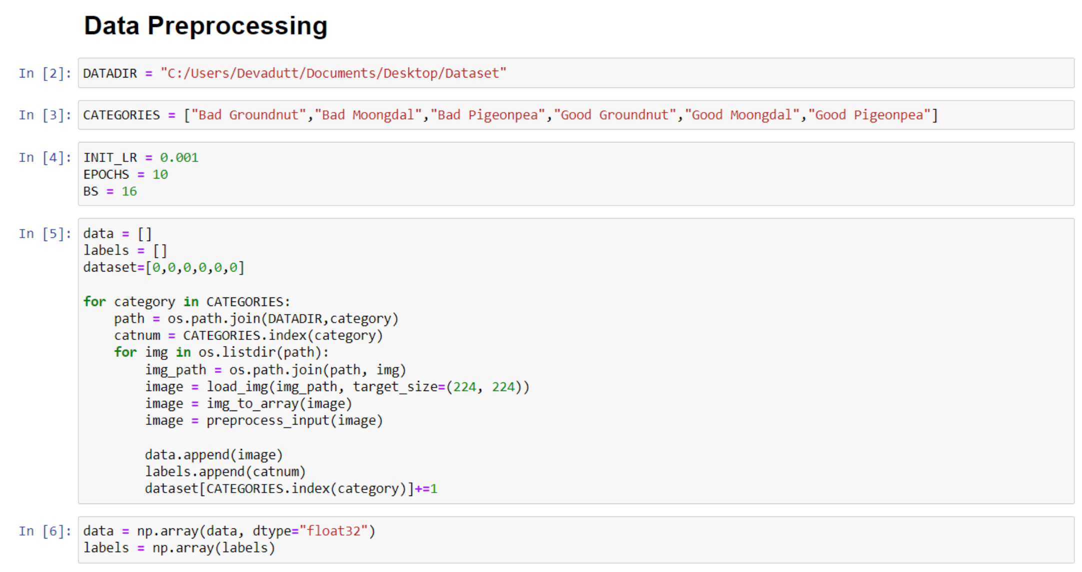


Fig 5.3: Images of three different seeds good and bad quality

5.2 Data Preprocessing

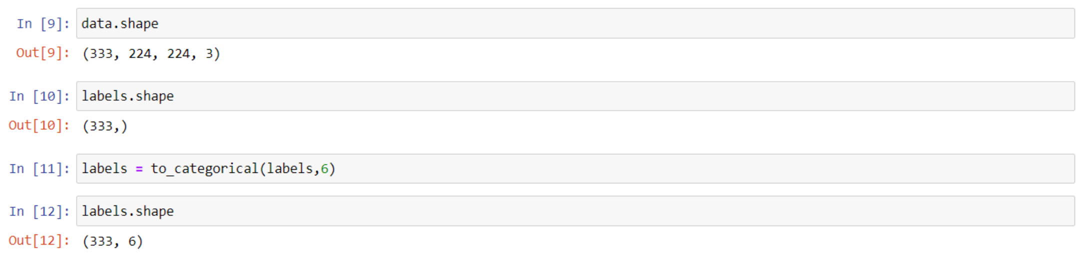
After selecting the images path, the images are passed through preprocessing operation, where the images are reshaped and converted to an array. All the images and their labels were appended to a list in order. Later the preprocessed lists are converted to a numpy array for easy processing in the next stages as shown in Fig 5.4. Label encoder is used for categorizing the labels numpy array for train and test split as shown in Fig 5.5.



Data Preprocessing

```
In [2]: DATADIR = "C:/Users/Devadutt/Documents/Desktop/Dataset"

In [3]: CATEGORIES = ["Bad Groundnut","Bad Moongdal","Bad Pigeonpea","Good Groundnut","Good Moongdal","Good Pigeonpea"]

In [4]: INIT_LR = 0.001
        EPOCHS = 10
        BS = 16

In [5]: data = []
        labels = []
        dataset=[0,0,0,0,0,0]

        for category in CATEGORIES:
            path = os.path.join(DATADIR,category)
            catnum = CATEGORIES.index(category)
            for img in os.listdir(path):
                img_path = os.path.join(path, img)
                image = load_img(img_path, target_size=(224, 224))
                image = img_to_array(image)
                image = preprocess_input(image)

                data.append(image)
                labels.append(catnum)
                dataset[CATEGORIES.index(category)]+=1

In [6]: data = np.array(data, dtype="float32")
        labels = np.array(labels)
```

Fig 5.4: Data Preprocessing

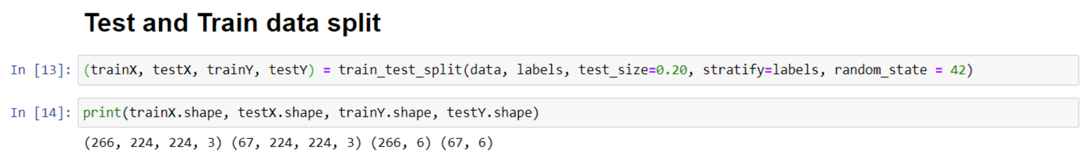


```
In [9]: data.shape
Out[9]: (333, 224, 224, 3)

In [10]: labels.shape
Out[10]: (333,)

In [11]: labels = to_categorical(labels,6)

In [12]: labels.shape
Out[12]: (333, 6)
```

Fig 5.5: Reshaping Numpy arrays

5.3 Train and Test Split

The numpy arrays are divided into two groups: test data and training data. A total of 20% of the data is set aside for model testing.



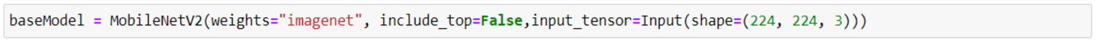
Test and Train data split

```
In [13]: (trainX, testX, trainY, testY) = train_test_split(data, labels, test_size=0.20, stratify=labels, random_state = 42)

In [14]: print(trainX.shape, testX.shape, trainY.shape, testY.shape)
         (266, 224, 224, 3) (67, 224, 224, 3) (266, 6) (67, 6)
```

Fig 5.6: Train and test split

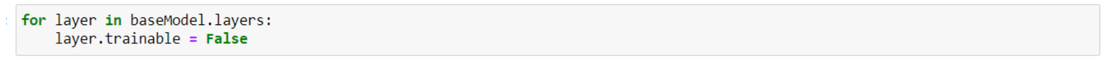
5.4 Model Creation

The creation of model starts with adapting the mobilenetv2 architecture as shon in Fig 5.7. Later the layers of this base model are freezed as shown in Fig 5.8 inorder to avoid destroying of the information they contain during future training stages. Now new layers are added on top of the frozen layers as shown in Fig 11(Average pooling layer, flatten layer, dense layer, dropout layer and softmax layer). This results in an updated version of mobilenetv2. Later as discussed in design part adam optimizer is used for model compilation as shown in Fig 5.10.



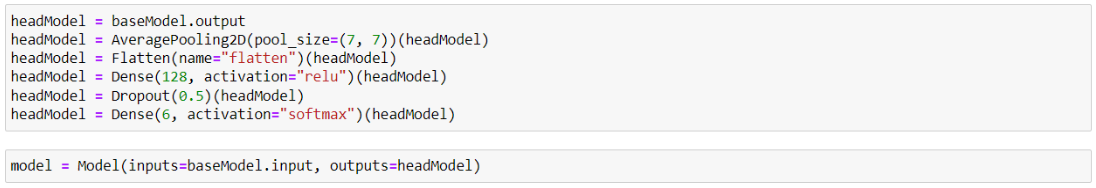
```
baseModel = MobileNetV2(weights="imagenet", include_top=False,input_tensor=Input(shape=(224, 224, 3)))
```

Fig 5.7: Importing pre-trained model



```
for layer in baseModel.layers:
    layer.trainable = False
```

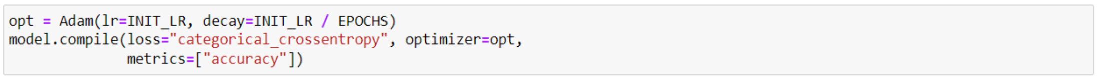
Fig 5.8: Freezing layers in the model



```
headModel = baseModel.output
headModel = AveragePooling2D(pool_size=(7, 7))(headModel)
headModel = Flatten(name="flatten")(headModel)
headModel = Dense(128, activation="relu")(headModel)
headModel = Dropout(0.5)(headModel)
headModel = Dense(6, activation="softmax")(headModel)
```

```
model = Model(inputs=baseModel.input, outputs=headModel)
```

Fig 5.9: Modification of the model



```
opt = Adam(lr=INIT_LR, decay=INIT_LR / EPOCHS)
model.compile(loss="categorical_crossentropy", optimizer=opt,
              metrics=["accuracy"])
```

Fig 5.10: Model compilation

5.5 Image Augmentation

As discussed in the design part for robustness of model, images are augmented with different variations as shown in Fig 5.11.Image Augmentation is applied to give the model different perspectives of similar image.

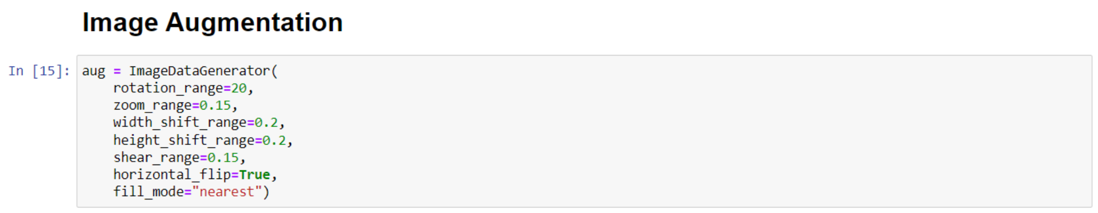
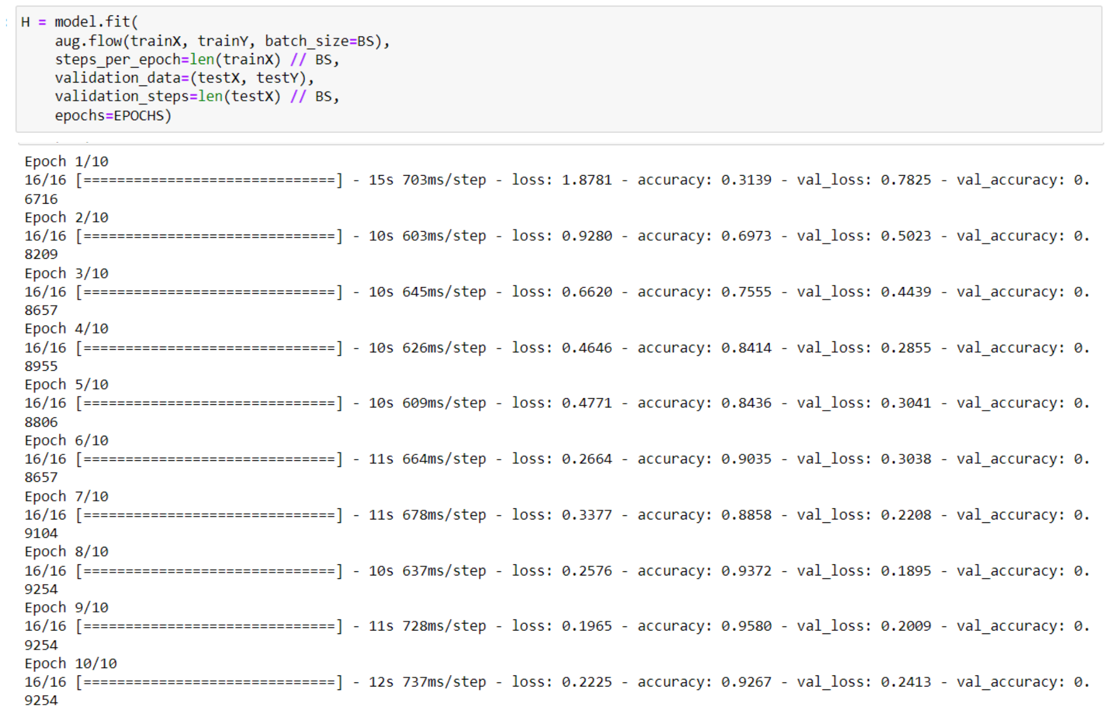


Image Augmentation

```
In [15]: aug = ImageDataGenerator(
             rotation_range=20,
             zoom_range=0.15,
             width_shift_range=0.2,
             height_shift_range=0.2,
             shear_range=0.15,
             horizontal_flip=True,
             fill_mode="nearest")
```

Fig 5.11: Image Augmentation

5.6 Model Training



```
H = model.fit(
    aug.flow(trainX, trainY, batch_size=BS),
    steps_per_epoch=len(trainX) // BS,
    validation_data=(testX, testY),
    validation_steps=len(testX) // BS,
    epochs=EPOCHS)
```

```
Epoch 1/10
16/16 [==============================] - 15s 703ms/step - loss: 1.8781 - accuracy: 0.3139 - val_loss: 0.7825 - val_accuracy: 0.
6716
Epoch 2/10
16/16 [==============================] - 10s 603ms/step - loss: 0.9280 - accuracy: 0.6973 - val_loss: 0.5023 - val_accuracy: 0.
8209
Epoch 3/10
16/16 [==============================] - 10s 645ms/step - loss: 0.6620 - accuracy: 0.7555 - val_loss: 0.4439 - val_accuracy: 0.
8657
Epoch 4/10
16/16 [==============================] - 10s 626ms/step - loss: 0.4646 - accuracy: 0.8414 - val_loss: 0.2855 - val_accuracy: 0.
8955
Epoch 5/10
16/16 [==============================] - 10s 609ms/step - loss: 0.4771 - accuracy: 0.8436 - val_loss: 0.3041 - val_accuracy: 0.
8806
Epoch 6/10
16/16 [==============================] - 11s 664ms/step - loss: 0.2664 - accuracy: 0.9035 - val_loss: 0.3038 - val_accuracy: 0.
8657
Epoch 7/10
16/16 [==============================] - 11s 678ms/step - loss: 0.3377 - accuracy: 0.8858 - val_loss: 0.2208 - val_accuracy: 0.
9104
Epoch 8/10
16/16 [==============================] - 10s 637ms/step - loss: 0.2576 - accuracy: 0.9372 - val_loss: 0.1895 - val_accuracy: 0.
9254
Epoch 9/10
16/16 [==============================] - 11s 728ms/step - loss: 0.1965 - accuracy: 0.9580 - val_loss: 0.2009 - val_accuracy: 0.
9254
Epoch 10/10
16/16 [==============================] - 12s 737ms/step - loss: 0.2225 - accuracy: 0.9267 - val_loss: 0.2413 - val_accuracy: 0.
9254
```

Fig 5.12: Model Training

Train data is used to train the model, and image augmentation is used on the train data. The model is trained for 10 epochs with 16 episodes in each epoch. Fig 5.12 shows the training outcomes of the model. Figure 5.13 depicts the training and validation accuracy. Figure 5.14 depicts the training and validation loss.

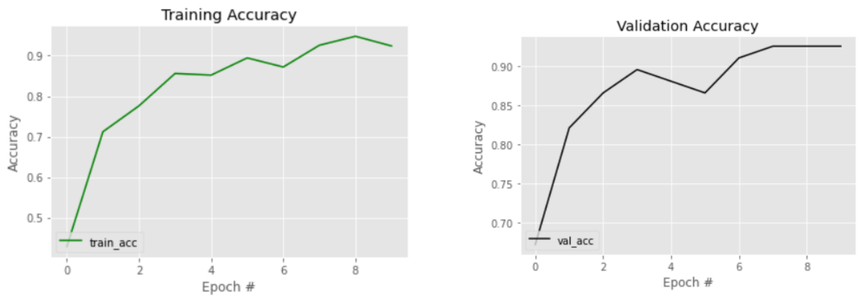


Fig 5.13: Training/Validation Accuracy

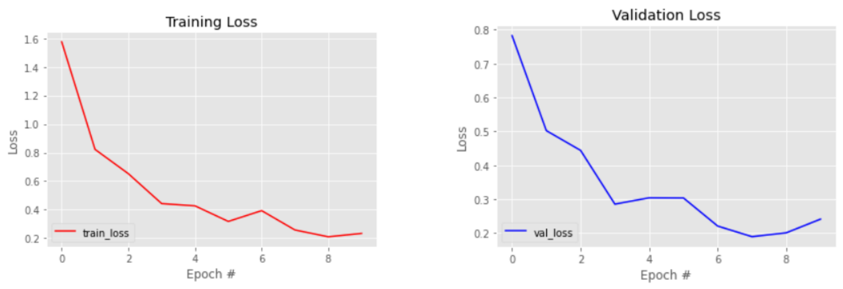
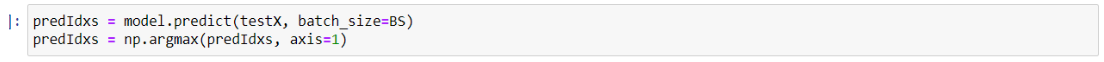


Fig 5.14: Training/Validation Loss

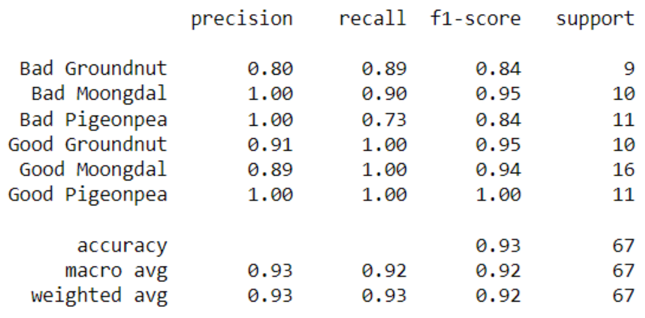
5.7 Model Testing

The deployed optimum model is used for testing as shown in Fig 16(a). Initially the model is tested on the test-data and it has achieved an accuracy of 93% as shown in Fig 16(b) and confusion matrix of testing can be seen in Fig 5.15.



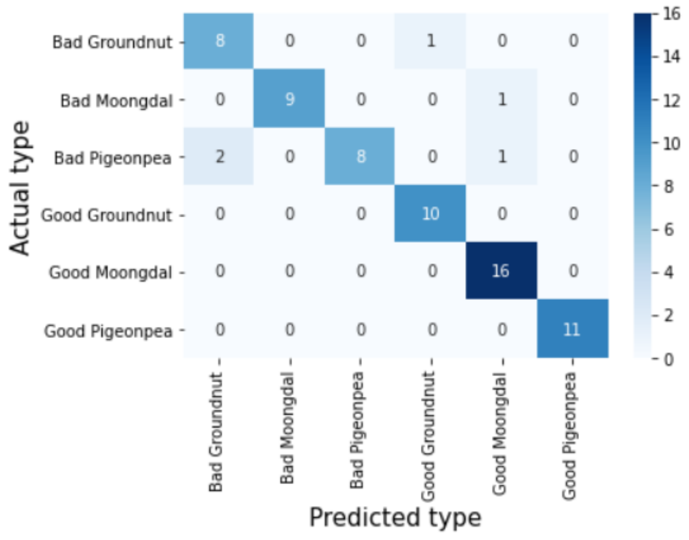
```
predIdxs = model.predict(testX, batch_size=BS)
predIdxs = np.argmax(predIdxs, axis=1)
```

Fig 5.15: Model Testing



```
               precision    recall  f1-score   support

 Bad Groundnut       0.80      0.89      0.84         9
  Bad Moongdal       1.00      0.90      0.95        10
 Bad Pigeonpea       1.00      0.73      0.84        11
Good Groundnut       0.91      1.00      0.95        10
 Good Moongdal       0.89      1.00      0.94        16
Good Pigeonpea       1.00      1.00      1.00        11

      accuracy                           0.93        67
     macro avg       0.93      0.92      0.92        67
  weighted avg       0.93      0.93      0.92        67
```

Fig 5.16: Classification report



	Bad Groundnut	Bad Moongdal	Bad Pigeonpea	Good Groundnut	Good Moongdal	Good Pigeonpea
Bad Groundnut	8	0	0	1	0	0
Bad Moongdal	0	9	0	0	1	0
Bad Pigeonpea	2	0	8	0	1	0
Good Groundnut	0	0	0	10	0	0
Good Moongdal	0	0	0	0	16	0
Good Pigeonpea	0	0	0	0	0	11

Fig 5.17: Confusion Matrix of testing

CHAPTER 6 – System Testing and Test Cases

The model generated at the end is used for predicting the labels for the class.

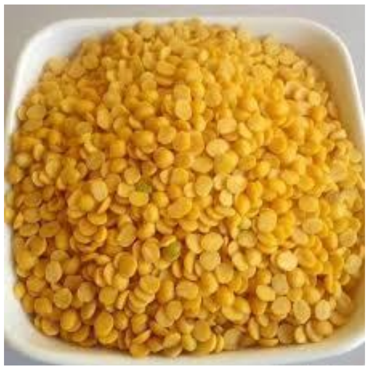
Expected Outcomes

The expected outcome of the model is to generate the class label to distinctly identify the images of seeds among the classes Bad Groundnut, Bad Moongdal, Bad Pigeonpea, Good Groundnut, Good Moongdal, Good Pigeonpea.

Outputs

The model is tested on new unseen data as part of 2nd phase of testing. The actual outputs that are generated for the unseen data are as follows:

Test case 1:

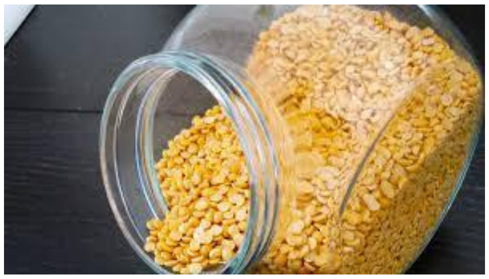


Actual : Good Pigeonpea

Predicted : Good Pigeonpea

Class	Bad Groundnut	Bad Moongdal	Bad Pigeonpea	Good Groundnut	Good Moongdal	Good Pigeonpea
Percentage	0.67	0.2	0.66	1.74	0.58	96.14

Test case 2:

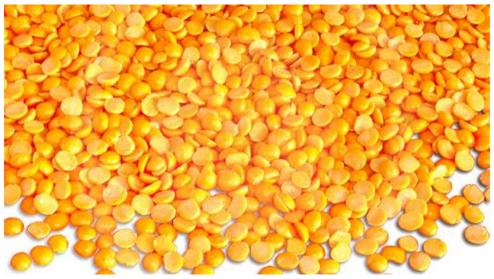


Actual : Good Pigeonpea

Predicted : Good Groundnut

Class	Bad Groundnut	Bad Moongdal	Bad Pigeonpea	Good Groundnut	Good Moongdal	Good Pigeonpea
Percentage	13.98	5.43	0.19	41.88	1.66	36.87

Test case 3:



Actual : Good Pigeonpea

Predicted : Good Pigeonpea

Class	Bad Groundnut	Bad Moongdal	Bad Pigeonpea	Good Groundnut	Good Moongdal	Good Pigeonpea
Percentage	0.15	0.12	0.10	4.50	0.18	94.96

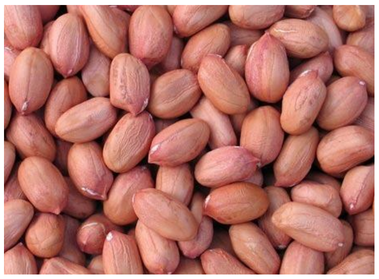
Test case 4:

Actual : Good Pigeonpea

Predicted : Good Pigeonpea

Class	Bad Groundnut	Bad Moongdal	Bad Pigeonpea	Good Groundnut	Good Moongdal	Good Pigeonpea
Percentage	0.42	0.29	0.21	14.88	0.56	83.64

Test case 5:

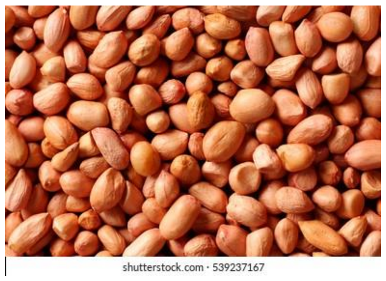


Actual : Good Groundnut

Predicted : Good Groundnut

Class	Bad Groundnut	Bad Moongdal	Bad Pigeonpea	Good Groundnut	Good Moongdal	Good Pigeonpea
Percentage	0.22	0.00	0.00	99.73	0.02	0.03

Test case 6:

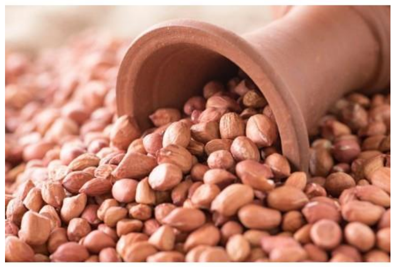


Actual : Good Groundnut

Predicted : Good Groundnut

Class	Bad Groundnut	Bad Moongdal	Bad Pigeonpea	Good Groundnut	Good Moongdal	Good Pigeonpea
Percentage	0.40	0.00	0.00	99.56	0.01	0.02

Test case 7:

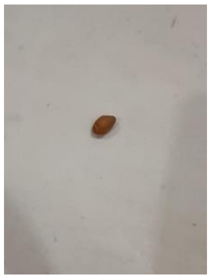


Actual : Good Groundnut

Predicted : Good Groundnut

Class	Bad Groundnut	Bad Moongdal	Bad Pigeonpea	Good Groundnut	Good Moongdal	Good Pigeonpea
Percentage	0.01	0.08	0.00	99.91	0.00	0.00

Test case 8:

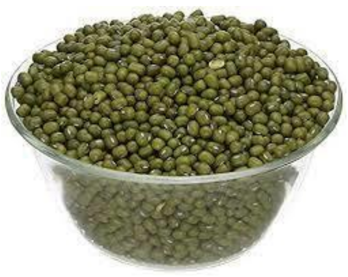


Actual : Good Groundnut

Predicted : Good Groundnut

Class	Bad Groundnut	Bad Moongdal	Bad Pigeonpea	Good Groundnut	Good Moongdal	Good Pigeonpea
Percentage	1.13	0.12	1.33	97.22	0.13	0.07

Test case 9:



Actual : Good Moongdal

Predicted : Good Moongdal

Class	Bad Groundnut	Bad Moongdal	Bad Pigeonpea	Good Groundnut	Good Moongdal	Good Pigeonpea
Percentage	20.12	4.95	2.72	6.68	61.53	4.01

Test case 10:

Actual : Bad Groundnut

Predicted : Bad Groundnut

Class	Bad Groundnut	Bad Moongdal	Bad Pigeonpea	Good Groundnut	Good Moongdal	Good Pigeonpea
Percentage	89.85	0.10	3.57	6.01	0.42	0.05

Test case 11:

Actual : Bad Groundnut

Predicted : Bad Groundnut

Class	Bad Groundnut	Bad Moongdal	Bad Pigeonpea	Good Groundnut	Good Moongdal	Good Pigeonpea
Percentage	75.94	0.07	5.30	18.59	0.06	0.03

Test case 12:

Actual : Good Moongdal

Predicted : Good Moongdal

Class	Bad Groundnut	Bad Moongdal	Bad Pigeonpea	Good Groundnut	Good Moongdal	Good Pigeonpea
Percentage	0.01	0.13	0.04	0.09	99.66	0.07

Test case 13:

Actual : Bad Pigeonpea

Predicted : Bad Pigeonpea

Class	Bad Groundnut	Bad Moongdal	Bad Pigeonpea	Good Groundnut	Good Moongdal	Good Pigeonpea
Percentage	4.50	0.06	95.19	0.16	0.09	0.01

Test case 14:

Actual : Good Moongdal

Predicted : Good Moongdal

Class	Bad Groundnut	Bad Moongdal	Bad Pigeonpea	Good Groundnut	Good Moongdal	Good Pigeonpea
Percentage	0.00	0.08	0.02	0.07	99.82	0.01

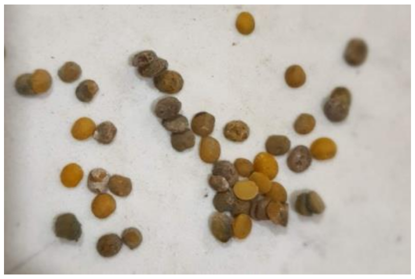
Test case 15:

Actual : Bad Pigeonpea

Predicted : Bad Pigeonpea

Class	Bad Groundnut	Bad Moongdal	Bad Pigeonpea	Good Groundnut	Good Moongdal	Good Pigeonpea
Percentage	4.50	0.06	95.19	0.16	0.09	0.01

Test case 16:

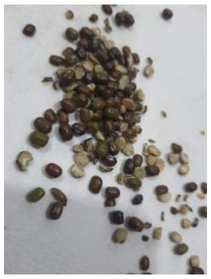


Actual : Bad Pigeonpea

Predicted : Bad Pigeonpea

Class	Bad Groundnut	Bad Moongdal	Bad Pigeonpea	Good Groundnut	Good Moongdal	Good Pigeonpea
Percentage	37.72	0.10	59.50	1.68	0.86	0.14

Test case 17:



Actual : Bad Moongdal

Predicted : Bad Moongdal

Class	Bad Groundnut	Bad Moongdal	Bad Pigeonpea	Good Groundnut	Good Moongdal	Good Pigeonpea
Percentage	0.00	99.84	0.00	0.02	0.14	0.00

Test case 18:

Actual : Bad Moongdal

Predicted : Bad Moongdal

Class	Bad Groundnut	Bad Moongdal	Bad Pigeonpea	Good Groundnut	Good Moongdal	Good Pigeonpea
Percentage	0.04	92.29	0.03	0.10	7.53	0.00

Test case 19:

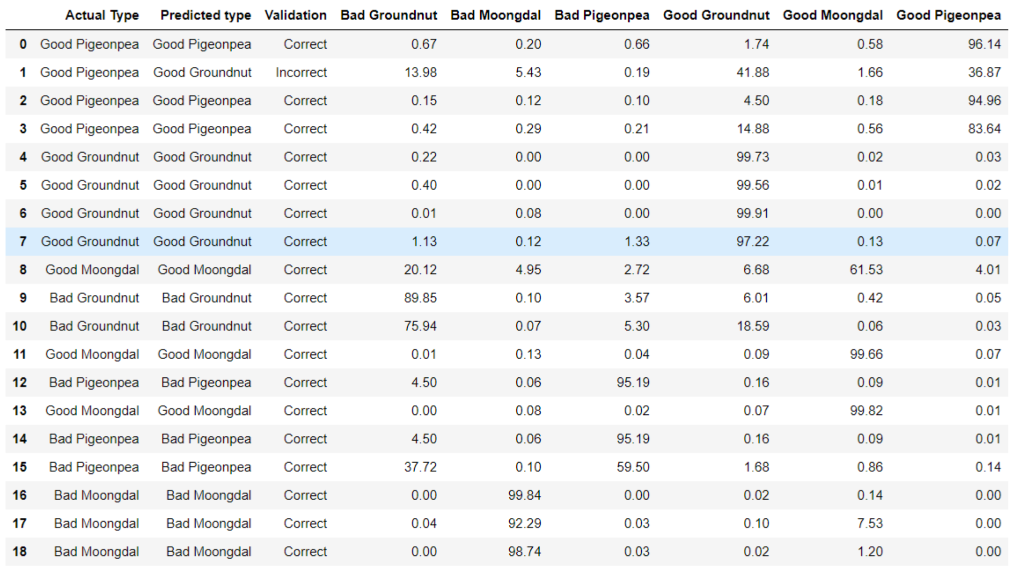
Actual : Bad Moongdal

Predicted : Bad Moongdal

Class	Bad Groundnut	Bad Moongdal	Bad Pigeonpea	Good Groundnut	Good Moongdal	Good Pigeonpea
Percentage	0.00	98.74	0.03	0.02	1.20	0.00

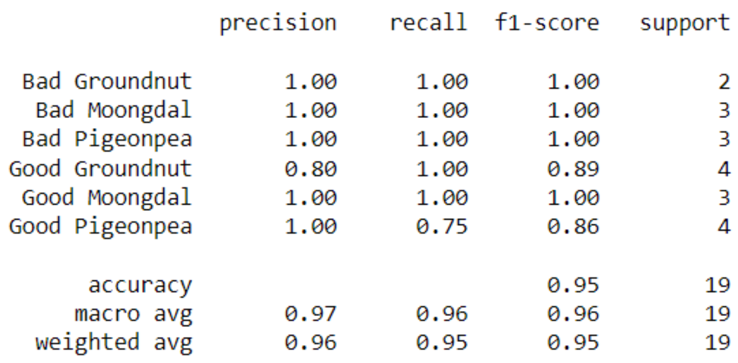
CHAPTER 7 – RESULTS AND DISCUSSION

The model is later tested on unseen new data and it has achieved an accuracy of 95%. Only 1 among the 20 images it predicted wrong so it can be found that the model is able to validate new dataset. The results of unseen data testing can be seen in Fig 7.1 and classification report and confusion matrix can be seen in Fig 7.2, 7.3 respectively.



	Actual Type	Predicted type	Validation	Bad Groundnut	Bad Moongdal	Bad Pigeonpea	Good Groundnut	Good Moongdal	Good Pigeonpea
0	Good Pigeonpea	Good Pigeonpea	Correct	0.67	0.20	0.66	1.74	0.58	96.14
1	Good Pigeonpea	Good Groundnut	Incorrect	13.98	5.43	0.19	41.88	1.66	36.87
2	Good Pigeonpea	Good Pigeonpea	Correct	0.15	0.12	0.10	4.50	0.18	94.96
3	Good Pigeonpea	Good Pigeonpea	Correct	0.42	0.29	0.21	14.88	0.56	83.64
4	Good Groundnut	Good Groundnut	Correct	0.22	0.00	0.00	99.73	0.02	0.03
5	Good Groundnut	Good Groundnut	Correct	0.40	0.00	0.00	99.56	0.01	0.02
6	Good Groundnut	Good Groundnut	Correct	0.01	0.08	0.00	99.91	0.00	0.00
7	Good Groundnut	Good Groundnut	Correct	1.13	0.12	1.33	97.22	0.13	0.07
8	Good Moongdal	Good Moongdal	Correct	20.12	4.95	2.72	6.68	61.53	4.01
9	Bad Groundnut	Bad Groundnut	Correct	89.85	0.10	3.57	6.01	0.42	0.05
10	Bad Groundnut	Bad Groundnut	Correct	75.94	0.07	5.30	18.59	0.06	0.03
11	Good Moongdal	Good Moongdal	Correct	0.01	0.13	0.04	0.09	99.66	0.07
12	Bad Pigeonpea	Bad Pigeonpea	Correct	4.50	0.06	95.19	0.16	0.09	0.01
13	Good Moongdal	Good Moongdal	Correct	0.00	0.08	0.02	0.07	99.82	0.01
14	Bad Pigeonpea	Bad Pigeonpea	Correct	4.50	0.06	95.19	0.16	0.09	0.01
15	Bad Pigeonpea	Bad Pigeonpea	Correct	37.72	0.10	59.50	1.68	0.86	0.14
16	Bad Moongdal	Bad Moongdal	Correct	0.00	99.84	0.00	0.02	0.14	0.00
17	Bad Moongdal	Bad Moongdal	Correct	0.04	92.29	0.03	0.10	7.53	0.00
18	Bad Moongdal	Bad Moongdal	Correct	0.00	98.74	0.03	0.02	1.20	0.00

Fig 7.1: Model prediction on unseen data



```
               precision    recall  f1-score   support

 Bad Groundnut       1.00      1.00      1.00         2
  Bad Moongdal       1.00      1.00      1.00         3
 Bad Pigeonpea       1.00      1.00      1.00         3
Good Groundnut       0.80      1.00      0.89         4
 Good Moongdal       1.00      1.00      1.00         3
Good Pigeonpea       1.00      0.75      0.86         4

      accuracy                           0.95        19
     macro avg       0.97      0.96      0.96        19
  weighted avg       0.96      0.95      0.95        19
```

Fig 7.2: Classification report on new data

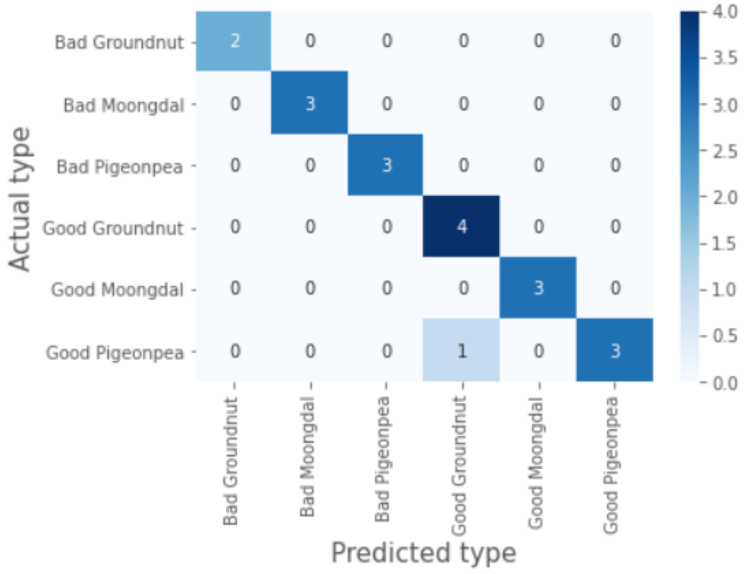


Fig 7.3: Confusion Matrix of unseen data

CHAPTER 8 – CONCLUSION AND FURTURE ENHANCEMENT

Conclusion:

This project proposed an efficient model for seed classification and quality detection. The model was trained for different epochs but the best model was achieved when epochs were 10. We have decided the epochs as 10 by observing the trend in training accuracy and loss while training the model as after 10 epochs the model is showing fluctuations.

Future Enhancement:

In Future work, we can train the model with huge number of seeds and implement the model in android application so that it would help people with less knowledge to classify and detect the quality of the seed

REFERENCES

1. “Jamuna, K.S.; Karpagavalli, S.; Vijaya, M.S.; Revathi, P.; Gokilavani, S.; Madhiya, E. Classification of Seed Cotton Yield Based on the Growth Stages of Cotton Crop Using Machine Learning Techniques. In Proceedings of the ACE 2010–2010 International Conference on Advances in Computer Engineering, Bangalore, India, 20–21 June 2010; pp. 312–315.”
2. “Qiu, Z.; Chen, J.; Zhao, Y.; Zhu, S.; He, Y.; Zhang, C. Variety identification of single rice seed using hyperspectral imaging combined with convolutional neural network. Appl. Sci. 2018, 8, 212.”
3. Kiratiratanapruk, K.; Temniranrat, P.; Sinthupinyo, W.; Prempree, P.; Chaitavon, K.; Porntheeraphat, S.; Prasertsak, A. Development of Paddy Rice Seed Classification Process using Machine Learning Techniques for Automatic Grading Machine. J. Sens. 2020, 2020, 7041310.
4. Luan, Z.; Li, C.; Ding, S.; Wei, M.; Yang, Y. Sunflower Seed Sorting Based on Convolutional Neural Network. In Proceedings of the ICGIP 2019 Eleventh International Conference on Graphics and Image Processing, Hangzhou, China, 12–14 October 2019; Pan, Z., Wang, X., Eds.; SPIE: Bellingham, WA, USA, 2020; Volume 11373, p. 129
5. Rozman, C. Denis Stajnko Assessment of germination rate of the tomato seeds using image processing and ˇ machine learning. Eur. J. Hortic. Sci. 2015, 80, 68–75
6. Taylor, J.; Chiou, C.P.; Bond, L.J. A methodology for sorting haploid and diploid corn seed using terahertz time domain spectroscopy and machine learning. AIP Conf. Proc. 2019, 2102.
7. Ali, A.; Qadri, S.; Mashwani, W.K.; Brahim Belhaouari, S.; Naeem, S.; Rafique, S.; Jamal, F.; Chesneau, C.; Anam, S. Machine learning approach for the classification of corn seed using hybrid features. Int. J. Food Prop. 2020, 23, 1097–1111.
8. Eli-Chukwu, N. Applications of Artificial Intelligence in Agriculture: A Review. Eng. Technol. Appl. Sci. Res. 2019, 9, 4377–4383
9. Agrawal, D.; Dahiya, P. Comparisons of classification algorithms on seeds dataset using machine learning algorithm. Compusoft 2018, 7, 2760–2765

10. Vlasov, A.V.; Fadeev, A.S. A machine learning approach for grain crop's seed classification in purifying separation. J. Phys. Conf. Ser. 2017, 803, 012177.
11. Maeda-Gutiérrez, V.; Galván-Tejada, C.E.; Zanella-Calzada, L.A.; Celaya-Padilla, J.M.; Galván-Tejada, J.I.; Gamboa-Rosales, H.; Luna-García, H.; Magallanes-Quintanar, R.; Guerrero Méndez, C.A.; Olvera-Olvera, C.A. Comparison of convolutional neural network architectures for classification of tomato plant diseases. Appl. Sci. 2020, 10, 1245
12. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. In Proceedings of the Advances in Neural Information Processing Systems 25 (NIPS 2012), Lake Tahoe, NV, USA, 3–6 December 2012; pp. 1097–1105.
13. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going Deeper with Convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
14. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the Inception Architecture for Computer Vision. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2818–2826
15. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778
16. Feature Detection and Analysis of Pigeon Peas Using Image Processing.
17. Xinshao, W. Weed Seeds Classification Based on PCANet Deep Learning Baseline. In Proceedings of the 2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference APSIPA ASC, Hong Kong, China, 16–19 December 2015; pp. 408–415.
18. https://www.tensorflow.org
19. https://numpy.org
20. https://matplotlib.org