Artificial Intelligence Enabled Instrumentation for Arecanut Segregation

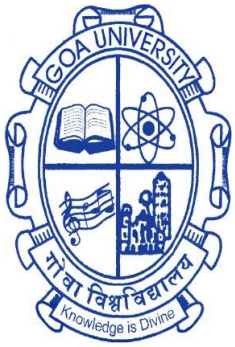
A Thesis submitted in partial fulfillment for the Degree of

DOCTOR OF PHILOSOPHY

in Electronics

in the School of Physical and Applied Sciences

Goa University



By

Sameer Manohar Patil

School of Physical and Applied Sciences

Goa University

October 2024

Artificial Intelligence Enabled Instrumentation for Arecanut Segregation

A Thesis submitted in partial fulfillment for the Degree of

DOCTOR OF PHILOSOPHY

in the School of Physical and Applied Sciences

Goa University

By

SAMEER MANOHAR PATIL

Research work carried out under

Prof. Jivan S. Parab (Guide)

and

Prof. Gourish M. Naik (Co-Guide)

School of Physical and Applied Sciences

Goa University, Taleigao Plateau

Goa – 403206

Examined
24/12/2024

October 2024

DECLARATION

I, Sameer Manohar Patil, hereby declare that this thesis represents work which has been carried out by me and that it has not been submitted, either in part or full, to any otherUniversity or Institution for the award of any research degree.

Place: Taleigao Plateau.
Date: 25-10-20224

Sameer Manohar Patil

CERTIFICATE

We hereby certify that the work was carried out under our supervision and may beplaced for evaluation.

(Prof. Jivan S. Parab)
Research Guide

(Prof. Gourish M. Naik)
Research Co-Guide

Dedicated

to

My parents

who always inspired me

"To fly high and taught me to never give up"

And

Padma Vibhushan (Late) Ratan Tata

for making

Bharat a proud nation

ACKNOWLEDGMENT

I thank **Almighty God** for the guidance, strength, and blessings throughout this journey, which made the completion of this thesis possible.

I would like to extend my heartfelt gratitude to my research guide, **Prof. Jivan S. Parab**, whose patience, understanding, and unwavering support has been pivotal in the completion of this thesis. Throughout this journey, his guidance, complete trust in my abilities, and thoughtful encouragement have allowed me to navigate challenges with confidence.

Special thanks to my mentor, **Prof. Gourish M. Naik**, who is a friend, guide, and philosopher, offering wisdom and encouragement at every step. Your insightful guidance and belief in my abilities have made this thesis possible. You have been incredibly generous with your time, always offering constructive feedback while allowing me the space to explore my ideas independently. I am deeply thankful for your mentorship, which has been a source of strength and inspiration throughout this research process.

I would like to thank my Vice Chancellor's nominees, **Prof. R. S. Gad** and **Dr. Ingrid Anne Nazareth,** for their feedback and suggestions during the Departmental Research Committee (DRC) meetings.

Special thanks to **Dr. Marlon Sequeira,** who always helped me when I was stuck with technical difficulties and assisted me throughout my Ph.D. work. I am deeply grateful to **Ms. Aparajita Naik,** who triggered me in the initial stage of my research, without which this milestone was not possible.

I will always be thankful to **Mr. Madhusudan Lanjewar, Dr. Aniket Gaonkar**, **Dr. Narayan Vetrekar, Mr. Agnel Lopes**, **Mr. Vishant Malik**, **Mr. William D'Souza**, **Ms. Ashwini Velip,** and **Ms. Pushpa** for their encouraging words, and constant motivation during my research work.

I am thankful to the **Management and Principal of DMs College and Research Centre, Assagao Goa**, for sanctioning one year of sabbatical leave and their support throughout, which greatly helped me in completing my work in the stipulated time.

I am grateful to **Mr. Andrew Karoff** for constantly assisting me with his technical hand in the design and development stage of my prototype model. I am thankful to **Mr. Arman Shaikh** for his assistance during my research work. Also, thankful to **Ms. Navami Parulekar** and **Mr. Sumedh Kitlekar** for their constant support and motivation.

My special thanks go to **Ms. Prashanti, Ms. Shanti, Ms. Minaxi, and Ms. Vidya** for encouraging me and taking **a** keen interest in my research work.

I am always thankful to **Mr. Tulshidas Patil**, **Mr. Khedekar**, and the **Support staff** of **Goa Bagayatdar** for assisting me in providing areca nut samples and giving details about

its segregation during my research work. Their courtesy during my visits is highly appreciated.

I am especially thankful to **Dr. Sachin Tendulkar** for his constant moral support and encouragement during my research work.

I extend my heartfelt thanks to **Prof. Rajnish Kamat** for motivating me to take on such a challenging topic and constantly monitoring the progress for its success. And **Dr. Sulaxana Vernekar** for continuously supporting and motivating me with her inspiring words.

This achievement would not have been possible without the unwavering support of my beloved wife, **Ramaa**, whose belief in me and constant encouragement inspired me to keep pushing forward for success.

I deeply appreciate the incredible patience and understanding of my son, **Mangirish**, throughout my PhD journey. His support, smiles, and boundless energy have been a source of joy and motivation, which helped me to stay focused and positive during challenging times.

Besides this, I thank all my family members and several other people who knowingly and unknowingly helped me in the successful completion of this thesis. It has been a fruitful and enjoyable experience, and the cooperation of all meant a lot to me.

TABLE OF CONTENTS

CHAPTER 2

CHAPTER 3

CHAPTER 4

CHAPTER 5

CHAPTER 6

CHAPTER 7

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AHE Adaptive Histogram Equalization

AI Artificial Intelligence

ANN Artificial Neural Network

AUC Area Under the Curve

CAGR Compound Annual Growth Rate

CED Canny Edge Detection

CIFAR Canadian Institute for Advanced Research

CLAHE Contrast Limited Adaptive Histogram Equalization

CNN Convolutional Neural Network

CPCRI Central Plantation Crops Research Institute

DCNN Deep Convolutional Neural Network

DL Deep Learning

DNN Deep Neural Network

DT Decision Tree

EEPROM Electrically Erasable Programmable Read Only Memory

FC Fully Connected

FN False Negative

FP False Positive

FPGA Field Programmable Gate Array

GDDR Graphics Double Data Rate

GDI Graphics Device Interface

GLCM Gray Level Cooccurrence Matrix

GLDM Gray Level Difference Matrix

GPIO General Purpose Input/Output

GRCM Gabor Response Co-occurrence Matrix

HDMI High Definition Multimedia Interface

HSI Hyperspectral Imaging

HSV Hue Saturation and Value

ICAR Indian Council of Agricultural Research

IoT Internet Of Things

KITTI Karlsruhe Institute of Technology and Toyota Technological Institute

kNN k-Nearest Neighbors

LBP Local Binary Pattern

LED Light Emitting Diode

LSTM Long Short Term Memory

LTP Long Term Potentiation

ML Machine Learning

MLP Multilayer Perceptron

MOT Multiple Object Tracking

NB Naive Bayes

NIR Near Infra Red

NLP Natural Language Processing

NN Nearest Neighbor

OSMF Oral Submucous Fibrosis

PCI Peripheral Component Interconnect

PWM Pulse Width Modulation

RF Random Forest

ReLU Rectified Linear Unit

ResNet Residual Neural Network

RMSprop Root Mean Square Propagation

ROC Receiver Operating Characteristic

RPM Rotations Per minute

RTC Real Time Clock

SDRAM Synchronous Dynamic Random Access Memory.

SGD Stochastic Gradient Descent

SGDM Stochastic Gradient Descent with Momentum

SMD Structured Matrix Decomposition

SSD Solid State Drive

SVM Support Vector Machine

TL Transfer Learning

TN True Negative

TP True Positive

USDA U.S. Department of Agriculture

UVLO Under Voltage Lock Out

VGG Visual Geometry Group

PREFACE

Agricultural farming is a critical sector for India's long-term economic prosperity. Around 70 % of rural households and 8% of urban households are involved primarily in agriculture for livelihood. Consumer dietary habits are changing, and agricultural commodity demand is increasing. The rising difficulties and opportunities necessitate a paradigm shift in agriculture's innovative industry. Among the different agricultural goods, Areca palm (Areca catechu L.), popularly known as Arecanut or Betel nut or commonly known as “Supari” in Indian households, has high religious ritual importance as well as commercial value due to its medicinal value. In recent times, the commercial value of the arecanut has increased many folds due to its multiple uses in mouth refresher products. Other important areas of its applications are digestive aid, anthelmintic, aphrodisiac, and to maintain stamina. Due to their antioxidant characteristics, arecanut extracts may be able to stop oxidative cell damage in normal cells. Studies reveal that anticancer compounds like flavonoid, polyphenol, carotenoid, selenium, vitamin C, and E, which showed chemotherapy and preventive activity, can be obtained from arecanuts. It is mostly grown in areas with an abundance of water resources such as monsoon rain regions, also in coastal and non-coastal areas of India. Across the world, arecanut is widely cultivated in Southeast Asia, South Africa, and the Pacific Ocean Islands.

Post harvesting, arecanuts are dried for about 45 days and are dehusked later. In Goa, these dehusked nuts are sold to traders. Goa Bagayatdar is one of the largest cooperative traders dealing with the purchase, segregation, and export of arecanuts in Goa. Normally, these nuts are grouped into seven classes viz. Supari, Laal, Vench, Tukada, and Baad, depending on the nut quality. However, it may be noted that the names of these classes and the number of classes may vary from region to region. To date, the segregation of these nuts into various classes is done manually by skilled labourers. However, due to the shortage of skilled labourers, the procedure of classification becomes a time-consuming one. This results in delayed payment and degrades the nut quality as they lie in the sack for a longer time. Further, depending on the skills of the labour, the classification of nuts may slightly vary from labour to labour leading to unfair payments. Therefore, there is a necessity to develop a segregation unit without human intervention that can be run in 24x7 mode. The thesis is organised into seven chapters.

The first chapter deals with the introduction of the topic and emphasises the need for automated segregation of arecanut. The chapter also deals with the importance of arecanuts in

the agricultural economy and their application in various fields of food product development. While the arecanut is a supreme offering to deities in Sanatan Customs, also, it is also considered auspicious to offer arecanut in religious invitations. The chapter also gives the various medicinal uses of arecanuts.

The second chapter is on review of the past work and objectives of the research work. This chapter gives details of the literature survey done in the classification of arecanuts. Huang K.Y. classified husked arecanuts into three classes using six geometric features, three colour features, and defects and used a back-propagation neural network classifier. He achieved a classification accuracy of 90.9 %. Patil S. et. al proposed an algorithm for pre-processing dehusked arecanut images. It was observed that Contrast-Limited Adaptive Histogram Equalization, Anisotropic Diffusion Edge Preserving Filter, and K-Means segmentation give the best results for applications involving arecanut segregation. Mallaiah S et al., in their proposed work on the classification of dehusked healthy and diseased arecanuts, have used LBP, Haar Wavelets, GLCM, and Gabor as texture features. An accuracy of 92.00% was achieved with the Gabor wavelet. Shabari S. B et. al proposed a binary classification of dehusked arecanuts based on their colour, texture, and density value. ANN outperformed other classifiers like logistic regression, kNN, NB classifiers, and SVM, achieving an accuracy of 98.8%. Patil S. et al. suggested utilizing CNN and MobileNet for binary classification of dehusked arecanuts, and they found that MobileNet performs better than CNN across all performance matrices. Siddesha S. et. al used the Nearest Neighbor (NN) classifier and applied feature extraction techniques like Wavelet, Gabor, Local Binary Pattern (LBP), Gray Level Difference Matrix (GLCM), and Gray Level Co-Occurrence Matrix for texture extraction of dehusked arecanuts into seven varieties. Using Gabor wavelet features, a classification rate of 91.43% was achieved. Jyothi K et. al proposed dehusked arecanut grading into five classes from Karnataka using LBP and SVM for feature extraction and grading, respectively. LBP feature extraction accuracy is 77%, while LTP feature extraction accuracy is 79%. Bharadwaj N. K et. al have classified dehusked arecanuts into four categories using LBP for extracting texture features. An image was divided into 2, 4, and 8 blocks, and from each of these blocks, the texture feature was extracted. Using the SVM classifier success rate of around 95% was achieved for 8 block of the image. Patil S. et. al have compared a custom CNN model against AlexNet for classifying dehusked arecanuts into five classes. They achieved an accuracy of 97.33% and 98.34% for 5 and 10 folds, respectively. It is observed that many researchers have studied the categorization of arecanuts

with husk. It is seen that a very little work is done on classifying dehusked arecanuts using Artificial Intelligence.

Also, In this chapter, we outlined our objectives of the research, problem statement, and solutions. At present, the segregation of the arecanuts is done manually by skilled labour at various collection centres of respective organisations. As described earlier due to the shortage of skilled labour, the segregation process is delayed. Due to the invention of AI, it is possible to use these technologies more resourcefully for nut segregation and make it user friendly with a design of self-explanatory GUI. Also, these technologies can lessen the cost of implementation drastically. In this regard, we plan to develop an AI-based algorithm to classify the arecanuts with an accuracy of above 95% and embed the algorithm in the instrumentation for automatic segregation. The main objectives of this study are a) Design of an image acquisition system for capturing quality images. b) Creating a database. c)Design of Deep Learning (DL) based algorithms. d)Development of AI based instrumentation for segregation and c) Performance evaluation of the system.

The third chapter is devoted to the methodology to investigate the research problem. The first part of the instrumentation development is arecanut image acquisition. A special imaging setup was for the feature extraction of all the varieties of arecanuts. In this arrangement, the sample is kept at the bottom and a camera at the top. The sample is located about 14 cm below the camera using a hollow cylinder with interior sides covered using black paper. The black material will prevent light from reflecting inside walls and remove the glaring at the lens. Around the camera, 20 white LEDs are arranged radially to illuminate the sample uniformly. For image acquisition, a 5MP Raspberry Pi-5 camera module is utilized, in conjunction with the Raspberry Pi 5 board via the MIPI camera serial interface protocol. An arecanut whose image is to be captured is placed on a black cloth at the bottom of the hollow cylinder. The hardware specifications used for the system were an Intel i5 processor having 32GB SDRAM and a NVIDIA GEFORCE RTX 3060 graphics card with 12GB DDR6 video memory.

The top variety of arecanut is called Supari class. This class of nut is the costliest among all and may have a tiny patch of husk at the top and a thin layer of husk at the bottom or sides. While dehusking or extra drying, fine cracks may develop. Next is the Laal variety has a hole at the bottom and sometimes may have minor cracks on the surface with a little husk at the bottom or sideways. They are followed by Tukada, which are pieces of arecanuts of good quality, Kharad variety has a husk on a significant portion of the nut. Due to improper drying, this husk remains on the nut and is primarily seen in nuts peeled using an

automatic peeling/dehusking machine. Since processing this class of nut in the industry requires additional steps, this class is priced below Tukada and Baad verities which has less weight, is uneven mainly in shape, and is porous. A dataset of 300 images with 60 from each class is created using the image acquisition setup.

The next step in the classification is Image Preprocessing. Here, a sequence of operations or manipulation is carried out on an image to improve the quality of the image, depending on the application. These operations on the image aim to reduce noise and distortion and enhance necessary features for further analysis and classification. Primarily, we have used three algorithms namely, contrast enhancement, image filtering, and image segmentation. We observed that CLAHE gives the best image contrast without deteriorating the image and maintaining good texture details. This is followed by image Filtering algorithms where in anisotropic diffusion edge preserving filter provides the best edge and texture preservation. Among the image segmentation algorithms K-means segmentation was able to segment all images in the database efficiently. The overall analysis shows that when we combine the above three techniques i.e. CLAHE, anisotropic diffusion edge preserving filter and K-means segmentation, we get the best possible results. These images are then cropped and fed to an AI based automated classification algorithm.

The next chapter 4 is on the classification algorithms used for segregation of arecanuts. Image The classification of the image was done using custom CNN, CNN-LSTM and various pre-trained DCNN models like MobileNet, AlexNet, GoogLeNet, SqeezeNet, and NasNetMobile. All these models were tried for various optimizers and different learning rates and these were tuned for suitable hyperparameters. The results of various pre-trained DL models were compared with the custom CNN-LSTM model. A learning rate of 0.0002 was found suitable after fine-tuning the model for various learning rates. Finally, the models were evaluated using various performance evaluation metrics.

The Chapter 5 is on results and analysis. In Binary classification when Custom CNN was compared with MobileNet, the later with augmentation gives the best results. However, when five-category classification was done, the CNN algorithm with 5-fold and 10-fold gives the best accuracy. It is also, observed that the customized CNN-LSTM model performs well compared to all other pre-trained DCNN models across all the optimizers and gives the best results for the RMSprop optimizer. When all the results were compared with all the metrics for different DL models across all the optimizers. It is seen that the standard deviation for CNN-LSTM with RMSprop optimizer is the minimum. The best accuracy of 99% was obtained using Custom CNN-LSTM and RMSprop optimizer on a standalone system.

Chapter 6 is on the Prototype Development Model. Developing an affordable and portable system for the segregation of five classes was a challenging task. The overall cost involved in developing a system for segregation into five classes is quite high due to the requirement of high speed and high ISO camera and GPU computational resources involved. Hence, it was decided after taking into views of the stakeholders that, Good and Bad class segregation could also ease the delay of farmer's payment. The simplified instrumentation required for two-class segregation would bring down the cost of development and make it more affordable to even small farmers. A complete assembly of segregator and various developmental details are given in the chapter.

A prototype model was developed using a Raspberry Pi 5 B single-board computer with a Broadcom BCM2711 processor working at 1.5GHz and 8GB RAM. The Raspberry Pi board is powered using an independent adapter. The details of the circuit schematic of the prototype model developed for the arecanut segregation is given in this chapter. The stepper motor coils and hammer solenoid are powered by 24V. The wheel indexing circuitry is powered by a 5V supply. As the stepper motors draw a heavy current of 6Amps at 24Volts, a power supply was custom-developed based on the switched mode principle, and the full details of the circuit diagram are given in this chapter.

Chapter 7 is the conclusion and future scope. In this research study, we have classified the five varieties of nuts viz. Supari, Laal, Tukada, Kharad and Baad. The approximate time taken for classification for a single fold is 27 minutes using the CNN-LSTM model. A high accuracy of 99% using a customized CNN-LSTM model with RMSprop optimizer was achieved. A prototype model was developed using a Raspberry Pi single board computer to extract good nuts from the heap to facilitate faster payments to the farmers. When the buyer has an idea about the percentage of good nuts, he is convinced about the amount to be paid to the farmers. The model developed gave an accuracy of near to 100 percent. The developed prototype can segregate approximately 120 Kg of arecanuts for a 10-hour operation.

As regards to the future scope, the author suggests various initiatives which one can try to improve upon the various aspects of the arecanut segregation unit. Such as using a weight cell to find the weight of each arecanut and select/reject them based on the density. He also refers to new arrivals of high computing platforms with AI instructions for PyTorch and Python built-in for increasing the speed of segregation. He also suggests the way and means to augment additional cameras for five-class arecanut segregation.

CHAPTER 1

INTRODUCTION

Homo sapiens first inhabited the globe 300,000 years ago in Africa, with the primary goal of survival being to find food. Earlier, Homo sapiens were hunters and food gatherers, the very reason for their nomadic lifestyle [1]. It was during the Neolithic revolution that agriculture emerged, which led to settled life. Great civilizations emerged on the banks of rivers, such as the Indus and Nile, the Indus Valley Civilization, and the Egyptian Civilization, respectively [2]. A settled lifestyle supported population growth, which nudged humans to increase the production of food grains and speed up the distribution process. It is human nature to improve the quality of living; thus, Man then conducted many experiments to improve the quality of food grain, fruits, and nuts in agriculture and plantation fields. Various methods were employed for the same, such as borrowing different varieties of seeds through the art of grafting- budding, cutting, and hybridization.

Industrialization marked the beginning of a new era in agriculture and plantation [3]. It reduced the burden of man through machines that were used for plowing, pulverizing, seeding, maneuvering, and harvesting. With modern technology, man is now exploring new areas and avenues to simplify tasks, reduce the required time, and improve the production, quality, and distribution of food grains, fruits, and nuts. Segregation of fruits and nuts is also seeing improvements from manual to automatization. This has been a boon as the previous approach saw a decrease in the quality of certain fruits and nuts when kept in abeyance for a more extended period (due to manual segregation) due to their very nature of being perishable.

Food scarcity in pre-independence India had significant consequences since agriculture was monsoon-dependent, and adverse rainfall and natural disasters resulted in crop failures. In independent India, the planning process designated agriculture as the most prioritized sector, emphasizing that 'everything can wait but agriculture. Despite the odds of unpredictable weather, diminishing soil health, rising atmospheric temperatures, and the rise of aggressive pests and pathogens, which persist post-independence, Indian agriculture has reached significant milestones, owing primarily to science-led agricultural progress. The most important milestone has been food security, which has instilled trust and improved the country's global profile. The tireless efforts of millions of our farmers, scientists, and planners have converted India from a

food deficit to a food surplus and net food exporter. Food grain production grew sixfold, from 51 million tons (Mt) in 1950/51 to more than 314 Mt in 2022 [4].

1.1 Importance of Arecanut

Among the various nuts cultivated in India, the arecanut has much significance due to its medicinal and economic significance. The arecanut, often known as betel nut, is the fruit of the Areca catechu palm [5,6]. The palm was originally prevalent in the Philippines, but it has been widely spread across the tropics by Austronesian migrations and trade from at least 1500 BCE due to its usage in betel nut chewing [7]. Arecanut has been used for centuries in conventional medicine, cultural customs, and social rites [8]. Despite its questionable health hazards when ingested regularly, it is recognized to have therapeutic characteristics that have been studied in numerous traditional medicine systems, including Ayurveda, Traditional Chinese Medicine, and Unani [9].

Arecanut contains a variety of biologically active compounds, like Alkaloids, Tannins, Polyphenols, Fatty acids, oils, etc [9]. All these components contribute to its medicinal properties. The alkaloid arecoline stimulates the central nervous system by acting on acetylcholine receptors, leading to increased cognitive function, enhanced focus, and mild euphoria [10]. Arecaidine and Guvacine have been studied for their potential to impact neurotransmitter activity. Guvacoline may have neuropharmacological effects. Condensed Tannins are known for their astringent properties and contribute to their traditional use in treating gastrointestinal issues [11]. Polyphenol compounds like Flavonoids and Catechins exhibit antioxidant activity, which can help neutralize free radicals and protect cells from oxidative damage. Fatty acids and oils in arecanuts have been linked to various health benefits, including antimicrobial activity [9]. Although modern medical research has questioned its safety, especially in habitual and long-term use, arecanut continues to be used in controlled dosages in traditional healing practices. Arecanut is often chewed for its stimulant effects, like caffeine. It is believed to enhance alertness, improve mental clarity, and increase energy. Fig 1 shows the various bioactive components of arecanut and their positive impacts targeting different organs. In Ayurveda, arecanut is used as a stimulant to relieve fatigue and improve digestion. It helps promote gas discharge and prevent bloating by inducing gastric secretions. The tannins in arecanut control intestinal motility and support digestive health. It is also prescribed for indigestion, constipation, and diarrhea [12,13]. In Unani medicine, it is similarly used for digestive issues. Arecanut has antimicrobial and antiparasitic properties and is historically used

to treat parasitic infections like tapeworms. The tannins and polyphenols in arecanut prevent the growth of bacteria and fungi, and it has anti-inflammatory qualities used for healing wounds and ulcers. In some cultures, arecanut paste is applied to skin wounds, and chewing it with betel leaves is believed to strengthen gums. However, regular arecanut chewing has been linked to oral cancer, raising concerns about its safety.

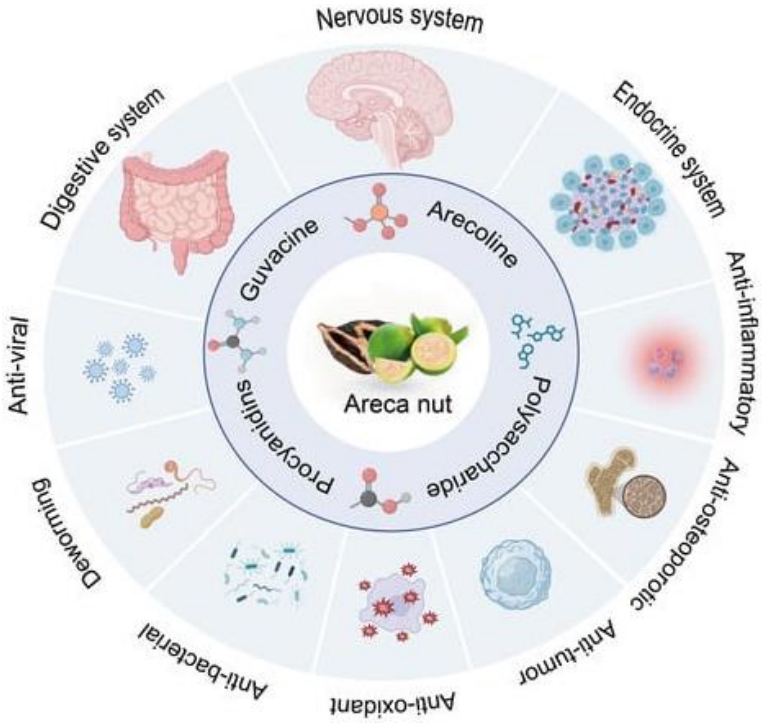


Fig 1.1: Medicinal significance of Arecanut [9]

Arecanut is traditionally used as a mild diuretic and laxative, promoting fluid excretion and relieving constipation. Arecoline, an alkaloid in arecanut, stimulates acetylcholine release, aiding memory and cognitive function, with potential in Alzheimer's therapy, though more studies are needed. It is also believed to enhance cognitive functions and mental clarity. Additionally, arecanut has vasodilatory properties, relaxing blood vessels, which may help lower blood pressure and improve circulation.

Research into arecoline shows that while it may have cardiovascular effects, excessive use raises concerns for heart health. Bioactive compounds in arecanut, such as polyphenols and flavonoids, have antioxidant properties that reduce oxidative stress, and some studies suggest hypoglycemic effects, though more research is needed. Arecanut has also been studied for anti-cancer properties, but its link to oral cancer, mainly due to habitual chewing, complicates its therapeutic potential. Chronic use has been strongly associated with oral submucous fibrosis (OSMF) and an increased risk of oral cancer. Arecoline is addictive, leading to dependency and

withdrawal symptoms, and excessive consumption can cause cardiovascular and neurological risks, outweighing its cognitive benefits.

1.1.1 Arecanut: Global Economic Scenario

Arecanuts hold significant importance globally, particularly in countries like India, Bangladesh, Sri Lanka, Myanmar, and Thailand, where they are widely cultivated.

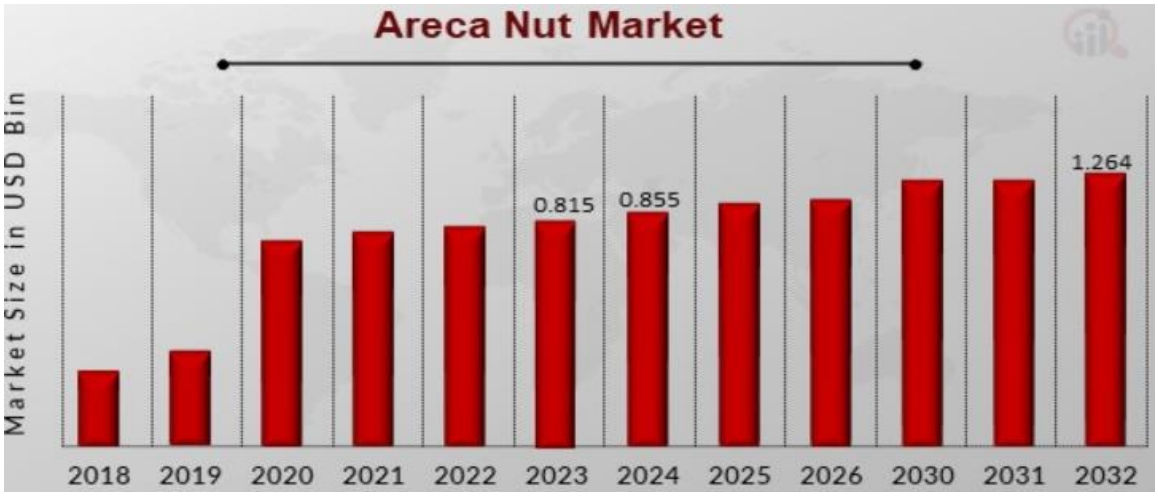


Fig 1.2: Arecanut market projection from 2018 to 2032

and consumed. As a major cash crop, arecanut supports the livelihoods of millions of farmers and workers involved in its cultivation, processing, and trade. The global demand for

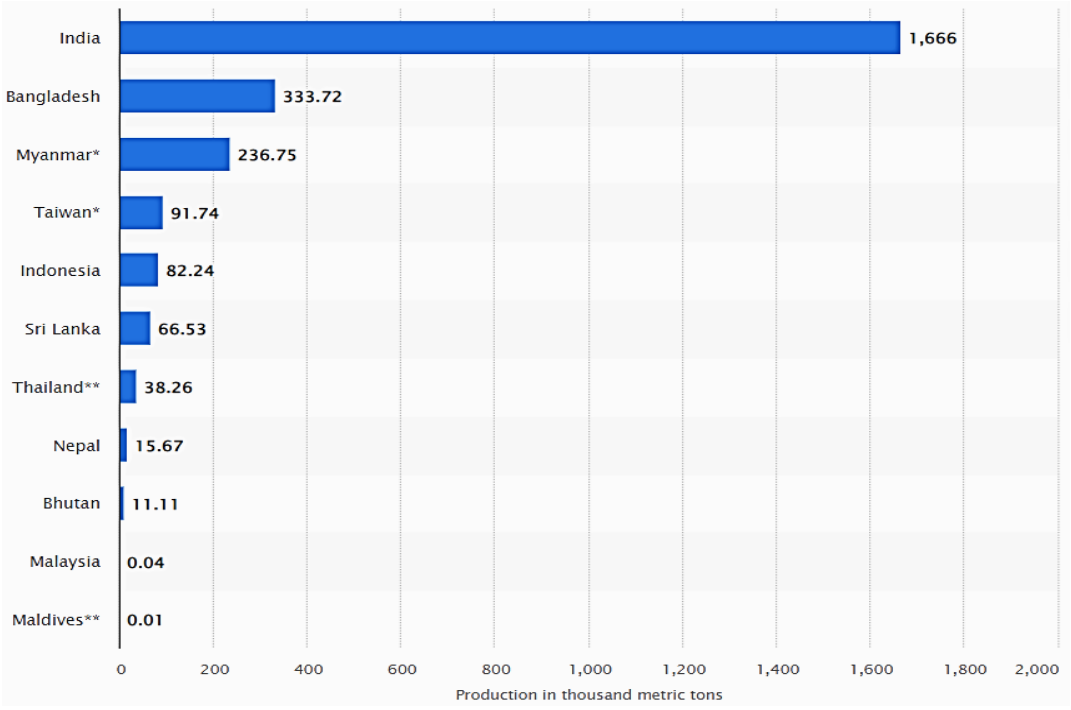


Fig 1.3: Arecanut production in Asia Pacific in 2022 (in 1000 metric tons)

arecanuts, especially for use in traditional practices and as a stimulant, drive its commercial value, making it an important export commodity in South and Southeast Asia. Arecanut

cultivation also contributes to rural economies by providing employment and income opportunities in agriculture and allied sectors. On a global scenario, as per Fig 1.2, the arecanut market is expected to proliferate at a compound annual growth rate (CAGR) of 5.00% from USD 0.85575 Billion in 2024 to USD 1.264 Billion by 2032 [14]. From Fig 1.3, it is seen that India is the highest producer of Arecanuts, with a total share of more than 50% of the global market [15].

1.1.2 Arecanut and Indian economy

Arecanut plays a crucial role in the Indian economy, particularly in rural regions where it is a significant cash crop. India is one of the largest producers and consumers of arecanut, with states like Karnataka, Kerala, Assam, and Tamil Nadu being key cultivation areas [16]. The crop supports the livelihood of thousands of farmers and labourers involved in its cultivation, processing, and trade. Arecanut is also a valuable export commodity, contributing to foreign exchange earnings. Despite its economic significance, concerns over health risks and regulations on its usage may impact its long-term financial sustainability.

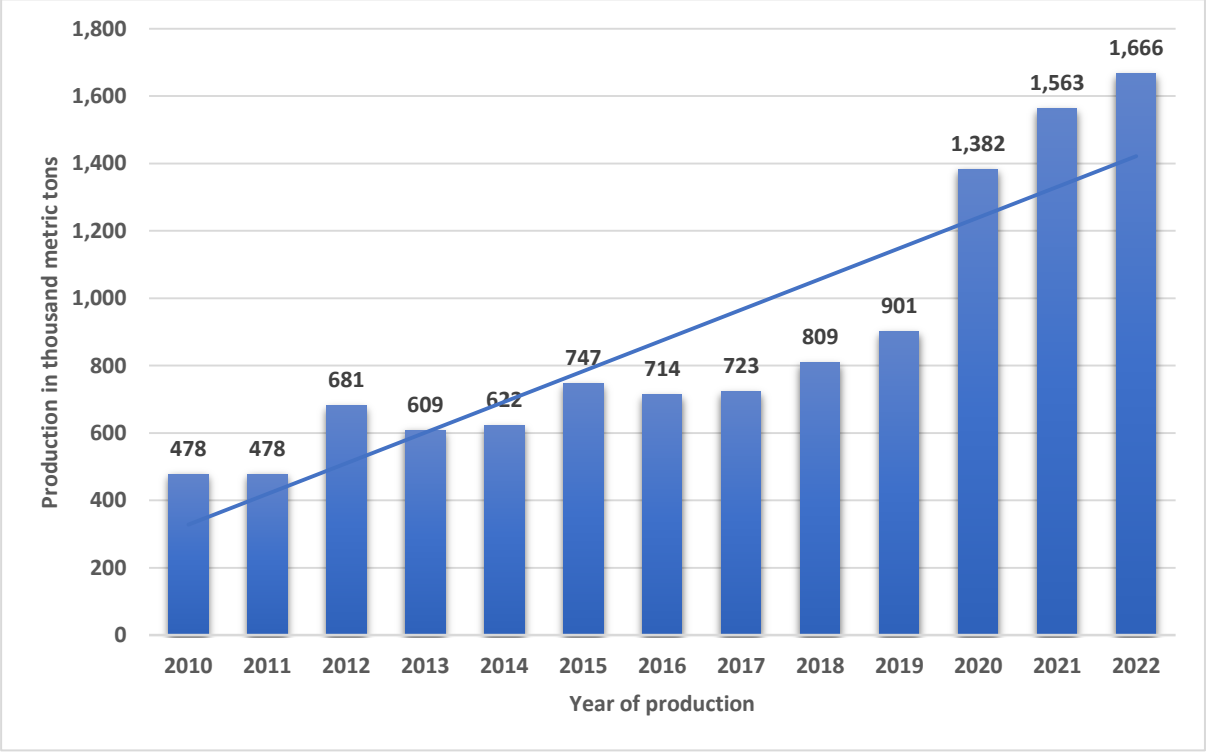


Fig 1.4: India's annual production of arecanuts from 2010 to 2022

Figure 4 shows India's yearly production of arecanuts in thousand metric tons from 2010 to 2022[17]. From 2010 to 2022, India's annual production of arecanuts steadily increased due to growing domestic demand and improved cultivation techniques. The states that predominately grow arecanut in India are Karnataka, Kerala, Andhra Pradesh, Assam, Meghalaya, Tripura,

Andaman and Nicobar Islands, Maharashtra, Goa, Tamil Nadu, Mizoram, West Bengal, and Pondicherry. Karnataka, Kerala, and Assam remained the top producing states, with Karnataka contributing the most significant share [18]. In 2010, India produced 478 thousand metric tons, which grew to over 1666 thousand metric tons by 2022.

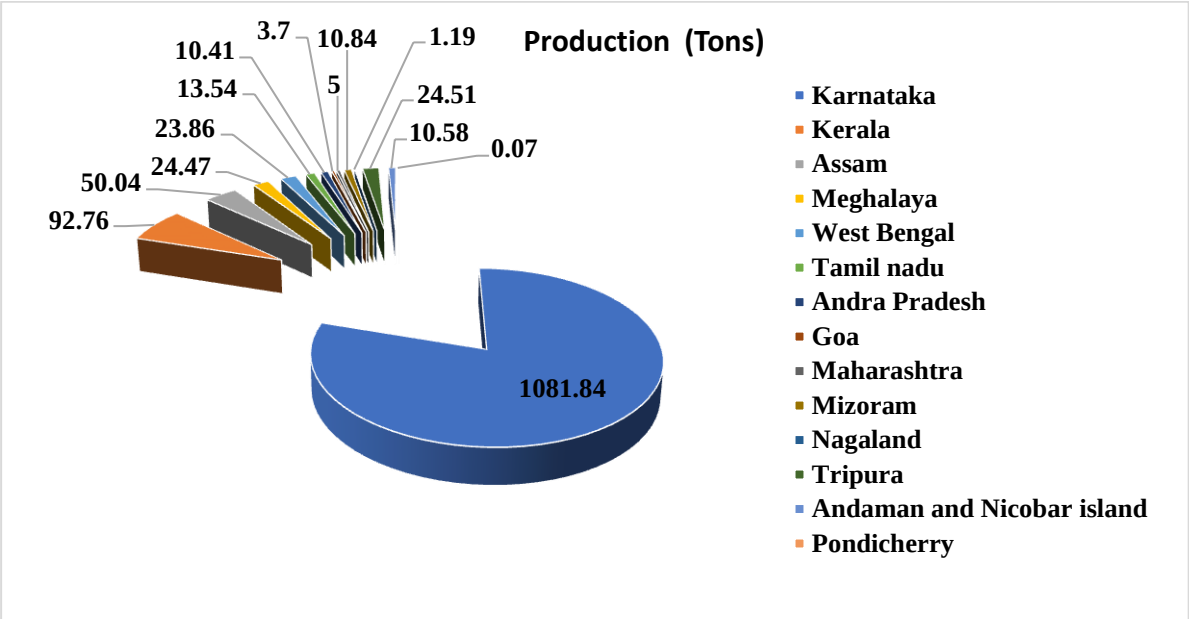


Fig 1.5: State-wise arecanut production in India for the year 2022

Figure 5 shows the state-wise arecanut production in India. India's statewide annual production of arecanuts is led by Karnataka, which accounts for a market share of 78.98% of the country's total output [18]. Kerala and Assam follow as significant producers, with both states contributing substantially to the national supply. Other states like Tamil Nadu, Maharashtra, and Meghalaya also contribute on a smaller scale. The favourable climate in these regions supports large-scale cultivation, making arecanuts a vital crop for rural economies.

1.1.3 Arecanut in Indian Traditions

Arecanut holds significant cultural and social importance in Indian traditions, especially in religious and ceremonial practices. It is commonly used in combination with betel leaves (paan) as a symbol of hospitality and respect, offered to guests during essential occasions such as weddings, festivals, and religious rituals. In Hindu traditions, the arecanut is often used as an offering to deities, symbolizing prosperity, and goodwill. It is considered auspicious and is included in many cultural practices across different regions of India.

1.2 Varieties of Arecanuts

India produces a wide range of arecanut varieties, each with distinct characteristics based on the region's climate, soil, and cultivation practices. Karnataka is known for its "Mangala" variety, which is highly regarded for its large size, smooth texture, and rich flavor, making it ideal for chewing. Kerala produces the "Sreemangala" variety, which is known for its superior quality and bright reddish-brown colour, making it popular in both domestic and international markets [19]. In Assam, the arecanuts have a slightly different texture and are often consumed locally in traditional forms like "tamol." Tamil Nadu and Maharashtra also produce unique varieties, which vary in size, shape, and taste, catering to regional preferences.

Arecanuts are categorized into different varieties based on size, quality, texture, and ripeness, each serving distinct applications. The high-quality, fully ripened varieties, like the "Mangala" from Karnataka and "Sreemangala" from Kerala, are primarily used for chewing with betel leaves (paan) in cultural and social practices. These nuts are preferred for their rich flavour and smooth texture [19]. Semi-ripe or moist varieties, such as those found in Assam, are commonly used in traditional chewing practices like "tamol," where the raw nut is consumed. Lower-grade or immature arecanuts are often processed for medicinal purposes, including digestive aids and traditional remedies. Due to their hardness and tannin content, some varieties are also utilized in industrial applications, such as in the production of dyes or as a component in certain herbal products [20]. This variety-based classification ensures that each type of arecanut is used efficiently, according to its properties and cultural significance.

1.2.1 Major Varieties of arecanuts

Wide Indian varieties of arecanut, such as Thirthahalli Local, South Canara Local (Karnataka), Kahikuchi, Nalbari (Assam), Shreewardhan, Maduramangala (Maharashtra), and Goa Local (Goa), are commonly grown by the farmers. Mangala (VTL-3) is the first improved variety introduced from China and adopted by many farmers. ICAR- Central Plantation Crops Research Institute (ICAR-CPCRI) Regional Station Vittal has evolved many enhanced varieties of arecanut as selections of hybrids of the VTL series. Secondary selection and inter se/sib mating are made to improve the Mangala variety to maintain purity. Mohitnagar is another high-yielding variety from Mohitnagar Centre of ICAR-CPCRI in West Bengal. Sumangala (VTL11), Swarnamangala (VTL-12), Sreemangala (VTL-17) and Shata Mangala (VTL-146) are other improved cultivars of arecanut. Hirehalli Dwarf is the dwarf mutant of arecanuts with short

stature. Hybrids VTL AH1 and VTL AH2 were developed using Hirehalli Dwarf as the female parent and improved tall varieties as the male parents [21].

1.2.2 Classification of arecanuts in Goa

In Goa, till the farmers started using machines for de-husking the areca nuts, white supari was classified as per the norms given. But, in the last 10 to 15 years, the nuts have been dehusked using machinery, and additional grades have come into existence.

Table 1.1 Goan Variety of arecanuts

Sr. No	Category of the nut	Characteristics
A	Baad	a. Fungus developed b. Holes all over the nut c. Irregular contour
B	Tukda	a. Piece of the nut
C	Vench	a. Big cracks on the body
D	Laal	a. A dark patch at the bottom/ A whole developed at the bottom
E	Kharad	a. White thick husk cover with fibres on the nut
F	Safed	a. A small cut at the top b. Minor cracks on the body (extra dry condition)
G	Supari	a. Clean body from the top b. A tiny patch of husk at the top c. White husk at the bottom

Fig.1.6: Classification of Areca nuts as followed by Goa Bagayatdar

The classification presently followed is given in Table 1. There are typically seven classes of areca nuts, as listed in Table 1.1, and their typical images are shown in Figure 1.6. and they are named a) Supari, b) Safed, c) Laal, d) Tukada, e) Vench, f) Kharad g) Baad, respectively [22].

Figure 1.6 contains an image of the “Supari” class nut. This class of nut is the costliest among all and has no defects. Behind this next class of nut is “Safed.” Safed arecanut is characterized by minor cracks or small cuts on the surface due to extra dryness. The next class of the nut is “Laal.” This nut has a hole at the bottom. This class nut is priced marginally below Safed class. The “Tukada” class is the next variety in pricing. These nuts are a broken variety of Supari and Laal class and are priced below them. The Tukada variety is mainly generated when nuts are peeled using an automatic de-husking machine. The fifth class is “Vench”. Here, one can find big cracks on the surface. “Kharad” class nut has a husk on a significant portion of the nut. Due to improper drying, this husk remains on the nut and is primarily available in nuts peeled using an automatic peeling/de-husking machine. Since processing this class of nuts in the industry requires additional steps, this class is priced below Vench. The last one, “Baad,” is the lowest class, as shown in the above figure. This nut has less weight, is uneven mainly in shape, and is porous [23].

1.3 Use of Technology in fruit segregation

New technologies for fruit segregation have significantly advanced the agricultural industry by making the sorting process faster, more accurate, and more efficient. Traditionally, fruit segregation was labour-intensive, relying heavily on manual sorting based on visual inspection for size, shape, colour, and surface defects. However, with the rise of automated systems, technology has taken over much of this work, offering precise and high-speed alternatives.

Modern sorting machines are equipped with sophisticated sensors, cameras, and computer vision systems that can evaluate multiple characteristics of fruits in real time [24,25]. For example, optical sorters use near-infrared (NIR) spectroscopy and colour imaging to assess external features like colour, size, and internal qualities, such as ripeness, sugar content, or even internal defects that are not visible to the naked eye [26,27]. This ensures that fruits with internal rot or under-ripeness are filtered out early, preventing them from reaching consumers and improving overall product quality [28].

Machine Learning (ML) and Artificial Intelligence (AI) have further enhanced fruit segregation by enabling systems to learn and adapt over time [29,30]. AI algorithms can process large amounts of data from fruit samples to refine sorting criteria, ensuring that the machines become more accurate in identifying subtle differences in fruit quality [31,32]. These systems can also

detect and classify defects such as bruising, discoloration, or irregular shapes that may not meet market standards.

1.3.1 Robotics in Fruit Segregation

Robotic arms and automated conveyor belts are increasingly being integrated into sorting lines, where they handle the movement of fruits through the sorting process [33]. These robotic systems can sort large volumes of fruit quickly and consistently, significantly reducing the time and labour required for manual sorting. By automating repetitive tasks, these systems also minimize human error, which can result from fatigue or subjective judgment.

In addition to improving efficiency and quality control, these new technologies are more sustainable. By accurately sorting fruits, they reduce food waste by ensuring that substandard fruits can be redirected to secondary markets, such as juices or processed foods, rather than discarded. Furthermore, optimized sorting helps farmers meet specific market demands, enabling them to fetch higher prices for their produce by delivering uniformly high-quality fruits [34]. New technologies in fruit segregation enhance productivity, improve product consistency, and support better use of resources, benefiting farmers, consumers, and the environment.

Fruit segregation is a crucial process in the agricultural supply chain, ensuring that only high-quality produce reaches consumers. Various methods are employed for fruit segregation, ranging from traditional manual sorting to advanced automated systems. In manual sorting, workers visually inspect and classify fruits based on size, colour, shape, and surface quality, often using tools like grading tables. However, this method can be time-consuming and prone to human error [35,36]. To enhance efficiency and accuracy, many producers are adopting automated techniques. Optical sorting machines utilize advanced imaging technology and sensors to assess fruits in real time, evaluating factors such as colour, size, and internal quality [34,37]. These systems can quickly identify and remove defective or under-ripened fruits. Additionally, machine-learning algorithms are increasingly being integrated into sorting processes, allowing for continuous improvement in classification standards based on data analysis. Other methods include the use of vibrating and air-jet systems that help to separate fruits based on size and weight [38]. Dry fruit segregation involves a series of methods to classify and sort dried fruits based on various quality parameters such as size, colour, texture, and moisture content. One of the primary methods is manual sorting, where workers visually inspect and categorize dried fruits, removing any defective or inferior quality pieces. While effective for small batches, this method can be labour-intensive and inconsistent for larger

quantities. Many producers utilize automated systems with advanced optical sorting technology to enhance efficiency. These machines use high-resolution cameras and sensors to analyze each fruit's characteristics in real time, allowing for precise classification based on predefined quality metrics [39, 40]. Additionally, air classifiers can be employed to separate dried fruits by weight and size, while vibratory screens help sort based on size through a series of mesh screens [41,42]. Moisture meters are also used to assess the moisture content of dried fruits, ensuring that they meet specific quality standards before packaging [43]. Since the fruit segregation technology is more mature and commercial systems are available, we have tried to use some of the sub-systems from these technologies for our system design and development.

1.3.2 Traditional methods of arecanut classification and segregation

Traditional arecanut classification and segregation methods in India are deeply rooted in local agricultural practices. After harvesting, the nuts are typically sun-dried and sorted manually by experienced farmers and labourers. The classification is primarily based on size, colour, shape, texture, and maturity. Larger, well-shaped nuts with a reddish-brown colour are considered of higher quality, often preferred for chewing purposes with betel leaves (paan). Smaller or irregular nuts, or those with a lighter colour, are set aside for different uses, such as medicinal applications or industrial processing. Arecanut segregation is essential for ensuring that only high-quality nuts reach the market.

1.3.3 Manual Sorting

Workers manually sort arecanuts based on visual inspection. They classify nuts by size, colour, shape, and surface quality, removing defective or inferior ones. While this method allows for detailed scrutiny, it can be labour-intensive and time-consuming, especially for large harvest.

Visual Inspection: Traditionally, the classification and grading of fruits and nuts were performed manually by farmers and workers who relied on visual inspection and tactile assessment. This method involved evaluating the size, shape, colour, and overall appearance of the produce.

Experience-Based Judgments: Skilled labourers with extensive experience could make relatively accurate assessments. However, these methods were subjective and prone to inconsistencies due to human error and fatigue.

1.3.4 Limitations

a. Subjectivity: The reliance on human judgment made consistency a challenge, as different workers might have different perceptions of quality.

b. Labour-Intensive: Manual sorting and grading were time-consuming and labour-intensive, limiting the scalability of operations.

c. Limited Criteria: Traditional methods often focus on a few visible criteria, neglecting internal quality factors such as taste, texture, and ripeness.

Dried nuts, commonly known as "supari," are widely used in traditional chewing, while semi-dry nuts, referred to as "chali," are processed differently. Region-specific varieties of arecanuts, such as those from Karnataka or Assam, are often distinguished for their unique flavour and texture, further adding to their classification. This traditional method ensures that arecanuts are sorted for various purposes, maintaining both the quality and economic value of the crop. Although labour-intensive, these techniques have been passed down through generations, preserving the cultural significance of arecanut farming in India.

Fig 1.7: Manual sorting of arecanut done by skilled labourers

1.3.5 Methods of Sale

The method of purchasing arecanuts in India has evolved over time, transitioning from traditional practices to more modern systems. Traditionally, arecanuts were primarily bought and sold in local markets, where farmers directly sold their produce to local traders or consumers.

The process was informal, with pricing based on regional demand, the quality of the nuts, and negotiations between buyers and sellers. These transactions often took place at weekly or seasonal markets, especially in rural areas, and were heavily reliant on personal relationships and trust.

In recent years, the purchase method has become more organized and formalized. With the rise of cooperatives, farmer groups, and online platforms, arecanuts are often sold through regulated auctions, mandis (wholesale markets), or directly to processing industries. Digital platforms and e-commerce have also made it easier for farmers to connect with buyers across the country, improving transparency in pricing and reducing the influence of agents [44]. Recent methods emphasize standardized quality checks, grading, and better logistics, providing farmers with better access to larger markets and fairer pricing mechanisms compared to traditional practices [45].

The commercialization of arecanuts is of significant importance for both the agricultural economy and the livelihoods of millions of people in India and other arecanut producing countries. It plays a vital role in ensuring the sustainable growth of the arecanut sector while driving rural economic development. As a major cash crop, particularly in states like Karnataka, Kerala, and Assam, the formal commercial market for arecanuts provides a stable income for farmers, labourers, and traders involved in its cultivation and processing [46]. Commercialization allows for better pricing structures, as arecanuts are traded through regulated markets, cooperatives, and auctions, ensuring farmers get fair market value for their produce. Moreover, commercialization enables the standardization of quality through grading and sorting, improving the product's appeal in both domestic and international markets. This, in turn, helps boost exports, contributing to foreign exchange earnings for the country. Beyond economic benefits, commercialization supports the development of industries reliant on arecanuts, including paan production, medicinal products, and even industrial applications like dyes. However, the process also requires addressing health-related concerns and balancing economic benefits with regulations aimed at minimizing the risks associated with excessive arecanut consumption.

1.4 Transition to Mechanical Systems

These systems help to sort arecanuts by size. Vibratory conveyors use oscillating movements to separate nuts based on weight and diameter, ensuring uniformity in the final product.

1.4.1 Grading Systems

Fig 1.8: Size based arecanut segregation unit

The agricultural revolution brought about the introduction of mechanical sorters that could separate fruits and nuts based on size and weight. These machines used conveyor belts, rollers, and shakers to facilitate the sorting process. Mechanical sorters significantly increased the efficiency of grading operations, reducing the reliance on manual labour and improving throughput.

1.4.2 Optical Sorting

Increasingly, producers are adopting automated systems equipped with optical sorting technology. These machines utilize high-resolution cameras and sensors to analyze each nut's characteristics in real time. They can detect imperfections such as discolouration, damage, and shape irregularities, allowing for rapid and precise classification.

By integrating these methods, arecanut producers can optimize the segregation process, enhance product quality, and improve market value, ultimately benefiting both farmers and consumers.

The use of AI (AI) in fruit segregation has transformed the agricultural sector by enhancing efficiency, accuracy, and speed in sorting processes [47]. AI-powered systems employ advanced algorithms and ML techniques to analyze various characteristics of fruits, such as size, colour, shape, and internal quality. Optical sorting machines equipped with AI can quickly process large volumes of fruit, identifying defects or ripeness levels that may not be visible to the naked eye [48].

AI systems utilize high-resolution cameras and sensors to capture detailed images of each fruit as it moves through the sorting line. These images are analyzed in real-time, allowing the system to classify fruits based on predefined quality metrics [49]. Furthermore, AI algorithms can learn from historical data to improve sorting accuracy over time [50]. By analyzing patterns and outcomes, these systems can adapt their criteria for segregation, resulting in continuous optimization of the sorting process. This not only enhances product quality but also increases

operational efficiency, as automated systems require less manual labour and reduce the chances of human error [51,52].

The integration of AI in fruit segregation not only improves the economic viability of fruit production by maximizing market value but also supports sustainable practices by minimizing waste and ensuring better quality control throughout the supply chain [47].

1.5 Machine Learning in Fruit Segregation

ML has emerged as a powerful tool in the agricultural sector, particularly in the field of fruit segregation. This technology leverages algorithms that enable computers to learn from and make predictions based on data without being explicitly programmed. In fruit segregation, ML algorithms analyze various characteristics of fruits, such as size, colour, shape, and ripeness, to classify them effectively. By training models on large datasets that include both high-quality and defective fruits, these algorithms can recognize patterns and make accurate classifications in real time [53].

The integration of ML in fruit segregation not only enhances the speed and efficiency of sorting processes but also significantly improves accuracy.

1.5.1 Instrumentation for segregation

Instrumentation plays a crucial role in effectively implementing ML in fruit segregation. It refers to the tools and devices used to measure and analyze various parameters of fruits during the sorting process. Advanced sensors, cameras, and imaging systems are essential components of modern fruit segregation systems [54]. These instruments capture detailed information about each fruit, including its dimensions, surface texture, and colour.

High-resolution cameras and multispectral imaging systems are often employed to gather data, while sensors can measure attributes like weight and firmness. The data collected by these instruments serve as input for ML algorithms, enabling them to analyze and classify fruits accurately. Moreover, the integration of Internet of Things (IoT) technology allows for real-time data transmission and monitoring, further enhancing the efficiency of the segregation process [55].

Together, ML and advanced instrumentation create a synergistic effect that revolutionizes fruit segregation, leading to improved quality, reduced labour costs, and better overall economic outcomes for producers.

Fig. 1.9: Goa Bagayatdar's latest technology used for cashew nut segregation.

1.6 Difficulties faced in manual segregation

Manual sorting of arecanuts faces several difficulties, primarily due to the labour-intensive and time-consuming nature of the process. Workers must visually inspect each nut to classify them based on size, colour, and quality, which is highly dependent on the experience and skill of the labourers. This approach is not only slow but also prone to human error, leading to inconsistencies in quality control. Additionally, the physical strain of sorting large quantities of arecanuts can lead to worker fatigue, reducing productivity over time.

Another major challenge is scalability. As demand for arecanuts grows, manual sorting becomes less feasible for large-scale production, as it requires a significant labour force, increasing operational costs. Seasonal fluctuations in labour availability also create bottlenecks during peak harvest times, further hindering the process. Overall, manual sorting is less efficient and cost-effective compared to mechanized or automated alternatives.

1.7 References

1. “Homo sapiens | The Smithsonian Institution’s Human Origins Program.” Accessed: Jul. 10, 2024. [Online]. Available: https://humanorigins.si.edu/evidence/human-fossils/species/homo-sapiens
2. “Indus Valley Civilisation,” *Wikipedia*. Jul. 06, 2024. Accessed: Jul. 10, 2024. [Online]. Available:https://en.wikipedia.org/w/index.php?title=Indus_Valley_Civilisation&oldid=1232900483
3. “Industrial Revolution,” *Wikipedia*. Jul. 21, 2024. Accessed: Jul. 24, 2024. [Online]. Available:https://en.wikipedia.org/w/index.php?title=Industrial_Revolution&oldid=1235926474
4. H. Pathak, J. Mishra, and T Mohapatra, *Indian-Agriculture-after-Independence*. Indian Council of Agricultural Research, New Delhi, 2022. [Online]. Available: https://icar.org.in/sites/default/files/2023-02/Indian-Agriculture-after-Independence.pdf
5. “Arecanut,” *Wikipedia*. Jul. 14, 2024. Accessed: Aug. 16, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Areca_nut&oldid=1234519837
6. “Areca catechu L., Sp. Pl. : 1189 (1753) | PALMweb.” Accessed: Aug. 18, 2024. [Online]. Available: https://www.palmweb.org/cdm_dataportal/taxon/5281044b-6737-4821-a487-af77b2298338
7. “Areca catechu L. | Plants of the World Online | Kew Science,” Plants of the World Online. Accessed: Aug. 18, 2024. [Online]. Available: http://powo.science.kew.org/taxon/urn:lsid:ipni.org:names:664107-1
8. P.-F. Liu and Y.-F. Chang, “The Controversial Roles of Arecanut: Medicine or Toxin?,” *IJMS*, vol. 24, no. 10, p. 8996, May 2023, doi: 10.3390/ijms24108996.
9. H. Sun, W. Yu, H. Li, X. Hu, and X. Wang, “Bioactive Components of Arecanut: An Overview of Their Positive Impacts Targeting Different Organs,” *Nutrients*, vol. 16, no. 5, p. 695, Feb. 2024, doi: 10.3390/nu16050695.
10. E. Badawy, K. Abouelsaoud, A. Ragab, and A. Kabbash, “Arecoline Biological Activity and Biotransformation: A review,” *Journal of Advanced Medical and Pharmaceutical Research*, vol. 0, no. 0, pp. 0–0, May 2023, doi: 10.21608/jampr.2023.201980.1050.
11. N. Z. T. De Jesus *et al.*, “Tannins, Peptic Ulcers and Related Mechanisms,” *IJMS*, vol. 13, no. 3, pp. 3203–3228, Mar. 2012, doi: 10.3390/ijms13033203.

12. M. Grover, “Areca catechu L. (Chikni Supari): A Review Based Upon its Ayurvedic and Pharmacological Properties,” *J Phytopharmacol*, vol. 10, no. 5, pp. 338–344, Sep. 2021, doi: 10.31254/phyto.2021.10510.
13. H. S. Cheema, M. P. Singh, “The Use of Medicinal Plants in Digestive System Related Disorders- A Systematic Review,” *J. Ayu. Her. Med.*, vol. 7, no. 3, pp. 182–187, Sep. 2021, doi: 10.31254/jahm.2021.7303.
14. M. R. F. https://www.marketresearchfuture.com, “Arecanut Market Demand, Size, Industry, Share, Growth, 2032.” Accessed: Oct. 06, 2024. [Online]. Available: https://www.marketresearchfuture.com/reports/areca-nut-market-12495
15. “APAC: arecanut production by country 2022,” Statista. Accessed: Oct. 06, 2024. [Online]. Available: https://www.statista.com/statistics/657902/asia-pacific-areca-nut-production-by-country/
16. “vikaspedia Domains.” Accessed: Oct. 06, 2024. [Online]. Available: https://vikaspedia.in/agriculture/crop-production/package-of-practices/plantation-crops/arecanut
17. “Trends of Arecanut production in India,” Tridge. Accessed: Oct. 06, 2024. [Online]. Available: https://www.tridge.com/intelligences/areca-nut/IN/production
18. “India production of ARECANUT.” Accessed: Oct. 06, 2024. [Online]. Available: https://agriexchange.apeda.gov.in/India%20Production/India_Productions.aspx?hscode=1092
19. “Arecanut Processing.pptx,” SlideShare. Accessed: Oct. 11, 2024. [Online]. Available: https://www.slideshare.net/slideshow/arecanut-processingpptx/253372268
20. N. R. Yanti, M. Andika, S. Maulida, Riani, I. Sulaiman, and N. M. Erfiza, “Utilization of arecanut (Arecha chatechu L.) extract for tannin based colorimetric indicator in smart packaging,” *IOP Conf. Ser.: Earth Environ. Sci.*, vol. 951, no. 1, p. 012057, Jan. 2022, doi: 10.1088/1755-1315/951/1/012057.
21. S. Bhat, V. Arunachalam, and P. V., “Production and Marketing scenarios of Arecanut in India.” Dr. Parveen Kumar The Director, ICAR-CCARI, Ela, Old Goa, Goa,2023.[Online] https://ccari.icar.gov.in/Extension%20Folder%20No.%20107.pdf
22. (2020) Goa Bagayatdar Bazar – One-stop-shop for all. In: Goa Bagayatdar Bazar – One-stop-shop for all. https://goabagayatdar.com/
23. S. Patil, A. Naik, and J. Parab, “Efficient Deep Learning model for de-husked Arecanut classification,” *JANS*, vol. 15, no. 4, pp. 1529–1540, Dec. 2023, doi: 10.31018/jans.v15i4.5067.

24. B. O. Olorunfemi, N. I. Nwulu, O. A. Adebo, and K. A. Kavadias, “Advancements in machine visions for fruit sorting and grading: A bibliometric analysis, systematic review, and future research directions,” *Journal of Agriculture and Food Research*, vol. 16, p. 101154, Jun. 2024, doi: 10.1016/j.jafr.2024.101154.
25. A. Bhargava and A. Bansal, “Fruits and vegetables quality evaluation using computer vision: A review,” *Journal of King Saud University - Computer and Information Sciences*, vol. 33, no. 3, pp. 243–257, Mar. 2021, doi: 10.1016/j.jksuci.2018.06.002.
26. K. K. Patel, A. Kar, S. N. Jha, and M. A. Khan, “Machine vision system: a tool for quality inspection of food and agricultural products,” *J Food Sci Technol*, vol. 49, no. 2, pp. 123–141, Apr. 2012, doi: 10.1007/s13197-011-0321-4.
27. R. Pandiselvam *et al.*, “Recent advancements in NIR spectroscopy for assessing the quality and safety of horticultural products: A comprehensive review,” *Front. Nutr.*, vol. 9, p. 973457, Oct. 2022, doi: 10.3389/fnut.2022.973457.
28. L. S. Magwaza, U. L. Opara, H. Nieuwoudt, P. J. R. Cronje, W. Saeys, and B. Nicolaï, “NIR Spectroscopy Applications for Internal and External Quality Analysis of Citrus Fruit—A Review,” *Food Bioprocess Technol*, vol. 5, no. 2, pp. 425–444, Feb. 2012, doi: 10.1007/s11947-011-0697-1.
29. I. Kutyauripo, M. Rushambwa, and L. Chiwazi, “Artificial intelligence applications in the agrifood sectors,” *Journal of Agriculture and Food Research*, vol. 11, p. 100502, Mar. 2023, doi: 10.1016/j.jafr.2023.100502.
30. “Advanced AI-driven Multi-Sensor Fusion for Fruit and Vegetable Segregation,” *IRJMETS*, Mar. 2024, doi: 10.56726/IRJMETS50912.
31. B. O. Olorunfemi, N. I. Nwulu, O. A. Adebo, and K. A. Kavadias, “Advancements in machine visions for fruit sorting and grading: A bibliometric analysis, systematic review, and future research directions,” *Journal of Agriculture and Food Research*, vol. 16, p. 101154, Jun. 2024, doi: 10.1016/j.jafr.2024.101154.
32. I. D. Apostolopoulos, M. Tzani, and S. I. Aznaouridis, “A General Machine Learning Model for Assessing Fruit Quality Using Deep Image Features,” *AI*, vol. 4, no. 4, pp. 812–830, Sep. 2023, doi: 10.3390/ai4040041.
33. M. D. D. Jangam, “Automation of Fruit Sorting Process Using Conveyor Belt System,” *IJRASET*, vol. 11, no. 6, pp. 2630–2635, Jun. 2023, doi: 10.22214/ijraset.2023.53976.

34. Z. Zhou *et al.*, “Advancement in artificial intelligence for on-farm fruit sorting and transportation,” *Front. Plant Sci.*, vol. 14, p. 1082860, Apr. 2023, doi: 10.3389/fpls.2023.1082860.
35. N. Mamat, M. F. Othman, R. Abdulghafor, A. A. Alwan, and Y. Gulzar, “Enhancing Image Annotation Technique of Fruit Classification Using a Deep Learning Approach,” *Sustainability*, vol. 15, no. 2, p. 901, Jan. 2023, doi: 10.3390/su15020901.
36. S. K. Behera, A. K. Rath, A. Mahapatra, and P. K. Sethy, “Identification, classification & grading of fruits using machine learning & computer intelligence: a review,” *J Ambient Intell Human Comput*, Mar. 2020, doi: 10.1007/s12652-020-01865-8.
37. pxdraft, “Fruit Sorting Machinery Market Size, Share & Trends [2031].” Accessed: Oct. 12, 2024. [Online]. Available: https://www.kingsresearch.com/fruit-sorting-machinery-market-726
38. P. Sarkar, “Use of shaking mechanism and robotic arm in fruit harvesting : A comprehensive review,” *J. Crop Weed*, vol. 17, no. 2, pp. 01–09, Feb. 2021, doi: 10.22271/09746315.2021.v17.i2.1444.
39. H. Tian, T. Wang, Y. Liu, X. Qiao, and Y. Li, “Computer vision technology in agricultural automation —A review,” *Information Processing in Agriculture*, vol. 7, no. 1, pp. 1–19, Mar. 2020, doi: 10.1016/j.inpa.2019.09.006.
40. S. K. Chakraborty *et al.*, “Development of an optimally designed real-time automatic citrus fruit grading–sorting machine leveraging computer vision-based adaptive deep learning model,” *Engineering Applications of Artificial Intelligence*, vol. 120, p. 105826, Apr. 2023, doi: 10.1016/j.engappai.2023.105826.
41. “Air classifier,” *Wikipedia*. May 22, 2022. Accessed: Oct. 12, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Air_classifier&oldid=1089267557
42. “Air Classifier Working Principle.” Accessed: Oct. 12, 2024. [Online]. Available: https://blog.praterindustries.com/air-classifier-working-principle
43. “Understanding Moisture Meters in the Food Industry: Ensuring Product Safety and Quality - HTA Instrumentation Pvt. Ltd.” Accessed: Oct. 12, 2024. [Online]. Available: https://htaipl.com/understanding-moisture-meters-in-the-food-industry-ensuring-product-safety-and-quality/
44. tss, “Marketing of Areca,” TSS Sirsi. Accessed: Oct. 12, 2024. [Online]. Available: https://www.tssindia.in/index.php/other1/about-arecanut-menu/marketing-of-areca

45. V. Mohanraj and R. Velusamy, “Analysis of marketing pattern of arecanut growers in Salem district of Tamil Nadu,” *Indian Journal of Extension Education*, vol. 58, no. 1, pp. 217–220, 2022, doi: 10.5958/2454-552X.2022.00038.X.
46. H. Jamanal and C. Murthy, “A Study on Growth of Procurement of Arecanut by Different Marketing Agencies for the Benefit of the Farmers,” *AJAEES*, vol. 41, no. 8, pp. 120–126, Jun. 2023, doi: 10.9734/ajaees/2023/v41i81988.
47. A. Taneja *et al.*, “Artificial Intelligence: Implications for the Agri-Food Sector,” *Agronomy*, vol. 13, no. 5, p. 1397, May 2023, doi: 10.3390/agronomy13051397.
48. K. Jha, A. Doshi, P. Patel, and M. Shah, “A comprehensive review on automation in agriculture using artificial intelligence,” *Artificial Intelligence in Agriculture*, vol. 2, pp. 1–12, Jun. 2019, doi: 10.1016/j.aiia.2019.05.004.
49. C. S. Nandi, B. Tudu, and C. Koley, “A Machine Vision Technique for Grading of Harvested Mangoes Based on Maturity and Quality,” *IEEE Sensors J.*, vol. 16, no. 16, pp. 6387–6396, Aug. 2016, doi: 10.1109/JSEN.2016.2580221.
50. “What Is Machine Learning (ML)? | IBM.” Accessed: Oct. 13, 2024. [Online]. Available: https://www.ibm.com/topics/machine-learning
51. M. Soori, B. Arezoo, and R. Dastres, “Artificial intelligence, machine learning and deep learning in advanced robotics, a review,” *Cognitive Robotics*, vol. 3, pp. 54–70, 2023, doi: 10.1016/j.cogr.2023.04.001.
52. D. B. Olawade *et al.*, “Smart waste management: A paradigm shift enabled by artificial intelligence,” *Waste Management Bulletin*, vol. 2, no. 2, pp. 244–263, Jun. 2024, doi: 10.1016/j.wmb.2024.05.001.
53. I. D. Apostolopoulos, M. Tzani, and S. I. Aznaouridis, “A General Machine Learning Model for Assessing Fruit Quality Using Deep Image Features,” *AI*, vol. 4, no. 4, pp. 812–830, Sep. 2023, doi: 10.3390/ai4040041.
54. B. O. Olorunfemi, N. I. Nwulu, O. A. Adebo, and K. A. Kavadias, “Advancements in machine visions for fruit sorting and grading: A bibliometric analysis, systematic review, and future research directions,” *Journal of Agriculture and Food Research*, vol. 16, p. 101154, Jun. 2024, doi: 10.1016/j.jafr.2024.101154.
55. P. Rajak, A. Ganguly, S. Adhikary, and S. Bhattacharya, “Internet of Things and smart sensors in agriculture: Scopes and challenges,” *Journal of Agriculture and Food Research*, vol. 14, p. 100776, Dec. 2023, doi: 10.1016/j.jafr.2023.100776.

CHAPTER 2

LITERATURE REVIEW & OBJECTIVES

In the agricultural and food industries, grading and classifying fruits and nuts are crucial procedures. These procedures guarantee quality control, improve marketability, satisfy client demands, and eventually ascertain the produce's economic worth. Standardization is dependent on precise and uniform grading and categorization methods at every stage of the supply chain, from manufacturing to sale. The goal of this chapter is to present a thorough overview of the literature on the methods, tools, and standards employed in the grading and categorization of fruits and nuts. This study aims to address issues in the sector, highlight technological developments, and identify best practices by synthesizing current research.

2.1 Adoption of Industrial Practices

2.1.1 Consistent Quality and Standards

Standardized criteria improved the consistency of grading, ensuring that consumers received produce of uniform quality. The establishment of grading standards by agricultural organizations and government bodies helped standardize the criteria used for classification and grading. For example, the USDA developed grades for various fruits and nuts based on size, colour, ripeness, and defect levels

2.1.2 Scaling Operations and Economic Impact

The use of mechanical systems enabled the scaling of operations to meet the demands of larger markets. Industrial-scale sorting and grading facilities emerged, particularly in regions with extensive fruit and nut production. The increased efficiency and consistency provided by mechanical systems contributed to the economic growth of the fruit and nut industries, enabling producers to meet the quality expectations of both domestic and international markets.

2.2 Modern Technological Advancements

Computer vision algorithms process the images to classify and grade products based on predefined criteria. This technology allows for rapid and accurate sorting, far surpassing the capabilities of manual inspection.

2.2.1 Artificial Intelligence and Automation

AI algorithms have revolutionized the classification and grading processes. These systems learn from large datasets to improve their accuracy and adapt to new types of produce. AI can predict internal quality factors such as ripeness and taste by analyzing external features and historical data, providing a more comprehensive assessment of quality. Robotics technology enables automated handling and sorting of fruits and nuts. Robotic arms equipped with grippers and sensors can carefully handle produce to minimize damage. Fully integrated automated systems combine optical sorting, AI algorithms, and robotics to create highly efficient grading lines that require minimal human intervention.

Modern systems offer real-time monitoring and data collection, allowing producers to track the quality of produce throughout the supply chain. Advanced traceability solutions ensure that each batch of fruits and nuts can be traced back to its origin, providing transparency and accountability in quality control.

2.3 Challenges and Future Trends

While modern systems are highly advanced, there are still limitations in detecting certain internal defects and quality factors. The high cost of advanced sorting and grading technologies can be a barrier for small-scale producers, limiting their adoption. Ensuring that the implementation of new technologies is sustainable and environmentally friendly remains a key challenge.

Ongoing research aims to improve the capabilities of AI and ML systems, making them more accurate and versatile. The integration of Internet of Things (IoT) devices with sorting and grading systems will enable more sophisticated monitoring and control. Innovations in sustainable technologies and practices will focus on reducing the environmental impact of sorting and grading operations. It is evident that the use of AI will not only speed up the process of classification and segregation but will also solve the problem of limited skilled human resources in the process of segregation.

2.4 Machine Learning Methods for Classification of Fruits and Nuts

Many researchers have attempted the classification and segregation of fruits and nuts using image processing, ML, and deep learning techniques. Below are some of the researchers who have worked in the classification of fruits and nuts using ML and deep learning techniques.

Md. Mahbubur Rahman et al., in their work on the detection and classification of local fruits through a web interface, utilized ResNet-50, VGG-19, Inception-V3, and MobileNet to achieve more detailed feature extraction. After collecting samples from different local places, image backdrops were removed. A total of 3240 samples of eight different fruits were collected from Bangladesh, including Carambola, Bilimbi, Elephant Apple, Emblica, Burmese Grape, Sapodilla, Tamarind, and Wood Apple. Using MobileNet for feature extraction achieved an accuracy of 99.21%[1].

Albarrak K et al. have worked on a Deep Learning-Based Model for Date Fruit Classification. In this work, a dataset of eight different classes of date fruit was created, and preprocessing techniques like image augmentation, decayed learning rate, model checkpointing, and hybrid weight adjustment were applied to increase the accuracy rate. The proposed model has also been compared with other existing models such as AlexNet, VGG16, InceptionV3, ResNet, and MobileNetV2. Using modified MobileNetV2 architecture, 99% accuracy was achieved[2].

Pham V-N et al. have worked on a Low-Cost, Deep-Learning-Based System for Grading Cashew Nuts by developing a module called SC3T that can be employed to integrate into the backbone of the YOLOv8 architecture. The proposed YOLOv8–Transformer model can accurately classify the cashew nuts into four quality grades. A dataset of 6000 images of different nuts in the middle regions of Vietnam was used for this work. The certainties of the predictions obtained were 90%, 97%, 98%, and 96% for “good” grade, “error 1” grade, “error 2” grade, and “error 3” grade [3].

Yurdakul, M. et al. have worked on the classification of four varieties of almonds with genetic designed lightweight CNN architecture using a public dataset. They used five light-weight CNN models viz. DenseNet121, EfficientNetB0, MobileNet, MobileNet V2, and NASNetMobile for classifying the almond images. The results of these five models were compared with the proposed Genetic CNN model, which has its hyperparameters determined by the Genetic Algorithm. This proposed model outperformed the other models with an accuracy of 99.55% [4].

Huang, B. et al., in their work on applications of ML in pine nuts classification, classified seven pine nut species using 210 near-infrared (NIR) spectra. They used five ML methods (Decision Tree (DT), Random Forest (RF), Multilayer Perceptron (MLP), Support Vector Machine (SVM), and Naive Bayes (NB to identify species of pine nuts. 303 images were used to collect morphological data to construct a classification model based on five convolutional neural

networks (CNN) models (VGG16, VGG19, Xception, InceptionV3, and ResNet50). The experimental results of NIR spectroscopy show the best classification model is MLP, and the accuracy is close to 0.99. Another experimental result of images shows the best classification model is InceptionV3, and the accuracy is close to 0.964. Four important ranges of wavebands, 951–957 nm, 1,147–1,154 nm, 1,907–1,927 nm, and 2,227–2,254 nm, were found to be highly related to the classification of pine nuts [5]. Sathish Kumar et al. have classified two pistachio species through neural embedding-based feature extraction and small-scale ML techniques. A total of 2148 images were used for this work. The model was evaluated using stratified random sampling. For feature extraction, deep neural network-based embeddings were used to acquire the vector representation of images. A success rate of 97.20% was obtained from Logistic Regression [6].

Singh Dilbag et al. have classified pistachio species using AlexNet, VGG16, and VGG19. A dataset of 2148 was used for the analysis. Success rates of 94.42%, 98.84%, and 98.14% were obtained from the AlexNet, VGG16, and VGG19, respectively [7]. Ünal Zeynep et al. have used 2094 images for discrimination of “whole kernel” hazelnut from three other classes viz “damaged kernel”, “shell”, and "undersized”. They used EfficientNetB0- EfficientNetB1- EfficientNetB2- EfficientNetB3 and InceptionV3 network structures for classification. An accuracy of 99.28 % was attained for the EfficientNetB2 and EfficientNetB3 models [8].

Emrah Dönmez et al. have classified three hazelnut varieties comprising 3627 images using big transfer deep learning models like (BiT)-M R50 × 1, BiT-M R101 × 3, and BiT-M R152 × 4. An accuracy of 99.49% was achieved with the BiT-M R152 × 4 model [9].

Rudresh Pillai et al. have used 2148 images for classifying two Pista species using a fine-tuned EfficientNetB3 model. An accuracy of 99.5% was achieved using the EfficientNetB3 transfer learning model [10].

Khaled Adil Dawood Idress et al. have classified three varieties of pistachio using two pre-trained models viz. VGG16 and Inception-V3 and a custom CNN architecture. A dataset of 2247 images was used for the analysis. Custom CNN outperformed the two pretained models with an accuracy of 97.02% [11]. Hakan Aktaş et al., in their study, have classified pistachios based on whether the nut is open or closed using deep learning techniques. The pre-trained AlexNet and Inception V3 models were used for testing the industrial data set and attained an accuracy of 96.13% and 96.54%, respectively [12].

Sivaranjani A. et al. have worked on computer vision-based cashew nuts grading systems using ML methods. The captured RGB images are normalized and converted into HSV colour

space. S channel from the HSV image is used for the segmentation process using the Otsu threshold technique. The constrained optimization-based feature selection method is used and 30 features are selected for further process. The SVM classifier is used for the classification, and the results obtained from different kernel functions are computed and compared. The 8-layer custom CNN developed with SVM 1-All provides an accuracy of 98.93% [13].

Mohammad Rahimzadeh et al. have done the detection and counting of open-mouth and closed-mouth pistachios using deep learning. A dataset comprising six videos with a total length of 167 s and 3927 labelled pistachios (9486 frames) was used for the study. The model is trained on the RetinaNet object detector network and later subjected to a new counter algorithm based on a new tracker to assign pistachios in consecutive frames and achieves an accuracy of 94.75% [14]. Jinu Lilly Joseph et al., in their work on the classification of fruits using deep learning, have used the Fruits 360 dataset containing 131 different classes of fruits and vegetables that can classify images into different categories. The model was developed using TensorFlow backend. Using this custom model, an accuracy of 94.35% was obtained [15].

Mimma N et al. have worked on the classification and detection of fruits using deep learning by combining a public data set FIDS-30 dataset of 971 images with 30 distinct classes of fruits and a private dataset comprising 761 images with eight categories of fruits. They have used YOLOv3 deep learning object detection algorithm for individual fruit detection across multiple classes, and ResNet50 and VGG16 techniques for the final classification for the recognition of a single category of fruit in images. An automatic fruit classification model with flask for the web framework was implemented, and an accuracy of 86% and 85% was achieved on the public dataset with ResNet50 and VGG16, respectively. Also, an accuracy of 99% with ResNet50 and 98% with the VGG16 model was obtained on the custom dataset [16].

Fu Y., et al have worked on grading methods for fruit freshness based on deep learning. They have utilized deep learning methods like ResNet-152, VGG-11, and GoogLeNet. AlexNet is chosen as the foundation network, while YOLO is used to extract the area of interest (ROI) from digital pictures. It was found that the four deep-learning models performed similarly. AlexNet outperforms $_{MSE}$ on the training set, but VGG outperforms MSE on the validation set [17]. Soleimanipour A et al. have classified four Iranian varieties viz. ‘Fandoghi’, ‘Kaleh-Ghouchi’, ‘Ahmad-Aghaei’, and ‘Akbari’ of pistachio nuts in bulk mode using EfficientNet deep learning model. The feature extraction block of EfficientNet-B3 was combined with a custom classification block. The classification accuracy of 98.00% was achieved using the EfficientNet model [18].

Zhang Y et al. have worked on the classification of fruits using a 13-layer DCNN model. They used data augmentation methods like image rotation, Gamma correction, and noise injection to enhance the dataset. In their work, they compared max pooling with average pooling and used Sgdm to train the developed CNN model with a minibatch size of 128. Using the above hyperparameters, an accuracy of 94.94% was attained. It was also observed that the use of data augmentation increases the overall accuracy of the model [19].

Taner, A. et al. have done a performance analysis of DCNN models for classifying 17 varieties of hazelnuts. A total of 250 images from each category totalling 4250, were captured using a 14.1-megapixel camera from 20cm. The performance of the custom CNN model was compared with the pre-trained models like VGG16, VGG19, ResNet50, and InceptionV3. The proposed CNN model provided an accuracy of 98.63% and outperformed all other pre-trained models [20].

Han Y et al. have worked on a non-destructive method using hyperspectral imaging (HSI), which is able to classify and estimate nut quality using deep-learning techniques. A dataset of 2300 images of *C. indicum* nuts was used for training the customized CNN model. The CNN network with the SVM classifier attained an accuracy of 93.48% for peroxide value estimation [21].

Panchbhai K et al. worked on small-size CNN(CAS-CNN) and modified mobileNetV2(CAS-MODMOBNET) to identify cashew nut and fruit diseases. A dataset created has four types of images viz disease fruit (65 images), disease nut (70 images), healthy fruit (32 images), and healthy nut (48 images). Data augmentation techniques like rotation range of 45 degrees, horizontal flip, and vertical flip were used to enhance the size of the dataset. The performance of the two custom models were compared with the pretrained models like DenseNet121, RestNet50, EfficientNetB0, VGG16, Xception, InceptionV3, NASNetMobile, MobileNetV2 and InceptionResNetV2. The CAS-MODMOBNET outperformed all other models by achieving an average accuracy of 99.8%. The CAS-CNN model, due to its fewer parameters (2,02,708), leads to a small size model making it efficient in terms of computational complexity. The smaller data set size may affect the generalizability of the models. Also, the authors have not specified the complexity of computation and requirements of resources for model deployment on various platforms [22]. Gautam V et al. have classified 2148 images of pistachio into two classes using a transfer learning-based optimized deep neural network based on its morphological features. Augmentation techniques like flipping, noising, and rotation are used to enhance the dataset. Various transfer learning models such as Inception, VGG16, SqueeezeNet, ResNet,

Xception, and VGG19 were used to test the performance. It is seen that the Xception model gives the highest accuracy of 98.63% among all the pre-trained models [23].

Rojas-Aranda et al., in their work on fruit classification and detection applications using deep learning, have used a picture categorization system based on lightweight Convolutional Neural Networks (CNN) to expedite the checkout process. A dataset with three types of fruits is used. Different input features, such as a single RGB colour, the RGB histogram, and the RGB centroid produced via K-means clustering, are introduced to the CNN architecture to improve classification accuracy. They attained an overall classification accuracy of 95% for fruits that were not in a plastic bag and 93% for fruits that were in a plastic bag [24].

Lisda L et al. in a study on the classification of two varieties of nuts, viz. kirmizi and sirt, achieved 96% and 86% accuracy based on two CNN models Inception V3 and ResNet50. In this work they have used a dataset containing 2148 images of pistachio [25]. Yonghua Yu et al. [13] used CNN-LSTM for real-time sorting of citrus fruit. A dataset of 270 images was used for this work. CNN is used for the detection of defective oranges whereas LSTM is utilized for predicting the position of oranges in the subsequent frame and categorical identification. An accuracy of 94.1 % was achieved using this novel method. Also, the error for path prediction was within 4.33 pixels 40 frames later. The average time to process a frame was between 28 and 62 frames per second [26].

2.5 AI-based Arecanut Classification Methods

Arecanut also has several distinct features that are crucial for its classification, grading, and marketability.

Size:

Diameter: The size of the arecanut can vary significantly, with the diameter being size category.

Shape:

Roundness: Arecanuts can be more or less round, and this shape characteristic affects their categorization.

Symmetry: Symmetrical arecanuts are generally preferred as they indicate uniform growth.

Colour:

Shell Colour: The colour of the outer shell can range from green to yellow to brown, depending on the ripeness and drying process.

Kernel Colour: The colour of the kernel (the inner part) can vary and is an indicator of quality and ripeness.

Surface Texture:

Smoothness: Smooth arecanuts are often preferred, as a rough surface may indicate damage or disease.

Cracks and Defects: The presence of cracks, holes, or other surface defects can lower the quality grade of the arecanut.

Weight:

Individual Weight: The weight of individual arecanuts is a key parameter for sorting and grading.

Moisture Content:

Surface Moisture: The moisture level on the surface of the arecanut affects its texture and durability during storage.

Fungal Infections:

Mold Presence: Visible mould or fungal growth indicates poor quality and can make the nut unsuitable for consumption.

Discoloration: Discoloration due to fungal infection is a common defect.

Insect Damage:

Bores and Holes: Damage caused by insects, such as boreholes, can significantly reduce the quality of the arecanut.

Larvae Presence: The presence of insect larvae is a serious defect.

Physical Damage:

Cracks: Cracks in the shell or kernel can be caused by mechanical damage during harvesting or processing.

Bruises: Bruising can occur due to handling and can affect the appearance and quality of the nut.

Uniformity:

Consistency: Uniform size, shape, and colour across a batch of arecanuts are preferred for commercial purposes.

Homogeneity: Homogeneity in physical and chemical properties is an indicator of high-quality arecanuts.

Storage Stability:

Shelf Life: The ability of arecanuts to retain their quality over time during storage is important for marketability.

Resistance to Pests: Arecanuts that are more resistant to pests and diseases are valued higher.

Understanding these features is essential for developing effective classification and grading systems, whether manual or automated. AI-based segregation units, for example, utilize these features to accurately sort and grade arecanuts, ensuring consistent quality and meeting market standards.

Few researchers have attempted to work on the classification and segregation of arecanuts by considering above mentioned parameters. The details of the same is given below:

Huang K.Y., has proposed the detection and classification of arecanuts with machine vision. In this method, he has used Neural Networks and Image processing techniques. He has utilized six geometric features, 3 colour features, and defects for the classification process. A back-propagation neural network classifier was used for sorting the quality of arecanuts. He achieved classification accuracy of 90.9%. [27].

Siddesha S. et. al, in their work on the texture-based classification of arecanut, have extracted different texture features from arecanut by applying approaches such as Wavelet, Gabor, Local Binary Pattern, Gray Level Difference Matrix, and Gray Level Co-Occurrence Matrix. To classify Nearest Neighbor classifier is used. 700 images of seven varieties of nuts were classified. Using Gabor wavelet features classification rate of 91.43% is achieved [28].

Mallaiah Suresha et al., in their proposed work on the classification of healthy and diseased arecanuts have used texture features of Local Binary Pattern, Haar Wavelets, GLCM, and Gabor. In the first stage, LBP was applied to each colour component of the HSI and YCbCr colour models, and an LBP histogram was formed. The statistical approach correlation is employed to quantify the distance between the histograms of the training set and the query sample, resulting in a success rate of 92.00%. In the next stage, texture features from Haar wavelets, GLCM, and Gabor were employed. RGB input is converted to HSI and YCbCr colour models, with texture details extracted from each colour component. Using kNN classifier 100% accuracy was achieved [29].

Ajit Danti et al. has converted RGB images into YCBCR colour space in their work on raw arecanut segmentation and categorization using three sigma control limits. They used a dataset of 629 images of two types of unpilled nuts. The effective segmentation of arecanuts is achieved using three-sigma control limits on the YCBCR image. A success rate of 98.97% for boiling nuts (BN) and 97.63% for non-boiling nuts (NBN) was achieved. The integration of three sigma control limits to segmentation and classification gives an organized technique for identifying various varieties of arecanuts. The segmentation and classification approach relies exclusively on colour components, which may be insufficient for more sophisticated grading criteria [30].

R D Meghana et. al have proposed the detection of different types of diseases in the arecanut such as mahili, stem bleeding yellow leaf spots using image processing using CNN. It uses an image as an input, assigns learnable weights to the objects in the image, and then learns the classification of the resulting images. A dataset of 241 images (diseased and healthy) was used for training and testing the CNN model. Categorical cross-entropy is employed in this case as a loss function, Adam is used as an optimizer function, and accuracy is used as metrics for model construction. To train the model 50 epochs were used and an accuracy of 93.3% was achieved for detecting the diseases in arecanut [31].

Dhanuja K C et. al have proposed Arecanut Disease Detection using Image Processing Technology. In this work, they have done texture-based grading of the arecanut. To extract several textural features from the arecanut, applications of Wavelet, Gabor, Local Bina ry Pattern, Gray Level Difference Matrix (GLDM), and Gray Level Cooccurrence Matrix (GLCM) are utilized. It makes use of the Nearest Neighbor (NN) algorithm. To demonstrate the effectiveness of the suggested approach, experiments were conducted using a dataset of 700 photos from 7 classes. An accuracy of 91.43% was attained using Gabor wavelet features [32].

N K Bharadwaj et. al have classified and graded arecanuts using texture-based block-wise local binary patterns for four classes of arecanuts from the Malnad region of Karnataka state. For feature extraction, they have used global texture features like the local binary pattern. They divided the image into k blocks in their research, and from each of those k blocks, the texture feature was extracted. They experimented with the complete image as well as individual blocks like 2, 4, and 8 using four classes of arecanuts. For grading of arecanut, they used a Support Vector Machine classifier. Using the SVM classifier success rate of around 95% was achieved for 8 block of the image [33].

Shabari Shedthi Billadi et. al have proposed a novel method for the classification of good and defective arecanuts based on their colour, texture, and density value using ML techniques. Artificial neural networks outperformed other classifiers like logistic regression, k-nearest neighbor, naive Bayes classifiers, and support vector machine. For better classification, density features are used. An accuracy of 98.8% was achieved using the proposed method [34].

Frencis M. S. et al. proposed maturity detection of husked arecanuts by applying HSV for feature extraction and K-NN for classification with k=1 and attained an accuracy of 87.42%. A dataset of 842 images was used and was divided into three classes as ripe, unripe, and old fruit [35]. Akshay S. et. al proposed a binary classification of husked arecanuts using the Otsu method to check the infected regions and a Decision tree classifier using the GLCM for the feature extraction method with an accuracy of 90% they also used background subtraction for removal of shadow effects found in the images [36].

Hegde A. et. al. have used ML techniques for identifying and categorizing diseases in arecanuts. For this work, a dataset containing more than 1100 images was created using healthy and unhealthy images of arecanuts, trunks, and leaves. The authors used a CNN-based model with binary cross entropy as the loss function and adam as optimizer. An accuracy of 93.05% was achieved using the proposed model [37].

Patil S. et. al proposed an algorithm for pre-processing of dehusked arecanut images. For this purpose, images were acquired using a Raspberry Pi 3 B+ board and a 5 Mega pixel Pi camera module. Eight methods for image preprocessing were explored, and it was observed that Contrast-Limited Adaptive Histogram Equalization, Anisotropic Diffusion Edge Preserving Filter, and K-Means segmentation give the best results for applications involving Arecanut segregation [38].

Mallikarjuna, S. et al., in their work on multi-type diseased arecanut image categorization using a CNN-based approach, have proposed a novel integrated method for multi-class (Healthy, Rot, Split, and Rot-Split) classification of arecanut pictures using deep convolutional neural networks and multi-gradient images. In this method, they have proposed to use multiple-Sobel masks for convolving with the input image. The masking operation assists in enhancing the fine details of the image despite images suffering from distortion due to disease infection. A dataset is created of 281 images and is enhanced to 10,116 images after the augmentation method (flip, blur, and rotation). These images are given as input to a combination of multi-gradient and AlexNet deep learning architecture. The proposed method outperforms in terms of classification rate, recall, precision, and F-measures [39].

K, Jyothi et. al have proposed a method for arecanut grading using ML methods. In this study, feature extraction is carried out utilizing the Local Binary Pattern, a global textural characteristic. Support vector machine classifier is used to grade arecanuts. LBP feature extraction accuracy is 77%, while LTP feature extraction accuracy is 79% [40]. Patil S. et al. utilized CNN and MobileNet for the binary classification of dehusked arecanuts. A dataset of 120 images was created using a special setup. Using CNN, an accuracy of 95% was achieved, whereas using a pre-trained MobileNet model, an accuracy of 100% was achieved [41].

Patil S. et al. have compared a custom CNN and AlexNet model for classifying de-husked arecanuts into five classes and achieved an accuracy of 97.33% and 98.34% for 5 and 10 folds, respectively. for this work a private dataset of 300 images was created using a special instrumentation setup. AUC-ROC curve was also for evaluating the custom CNN model [42]. Chandrashekhara H. et al. have classified healthy and diseased arecanuts using SVM Classifier. The images are segmented using the Structured Matrix Decomposition Model (SMD), and the LBP features are retrieved using the SVM classifier. Using above method an accuracy of 98% was achieved [43].

Suresha M et al have developed a novel method for arecanut classification based on texture features. A Gabor Response Co-occurrence Matrix (GRCM) is constructed and classification is done using kNN and Decision Tree (DT) classifier based on GRCM features. There are twelve Gabor filters are designed (Four orientations and three scaling) to capture variations in lighting conditions. The kNN classifier has given a success rate of 75.64. A decision tree is used for classification, and got a classification accuracy of 95.69% for the Twoing rule, 95.74% for GDI, and 96.31% for Entropy [44].

By consolidating and analyzing the current body of knowledge on the classification and grading of fruits and various types of nuts, including arecanuts, this literature review aims to provide a detailed understanding of the methodologies and technologies used in these processes. It highlights best practices, technological advancements, and the challenges faced by the industry, offering valuable insights for researchers, policymakers, and industry practitioners. This comprehensive overview will support the development of more efficient and accurate classification and grading systems, ultimately enhancing the quality and marketability of fruits and nuts.

The literature review also reveals that many scholars have studied the categorization of arecanuts with husk. It is noticed that very little work is done on classifying de-husked Arecanuts using deep learning techniques.

2.6 Aims and Objectives

The main aim of the research work is to design and develop a farmer friendly arecanut segregation machine incorporating the latest AI tools and techniques. In this chapter, we have outlined our objectives of the research and it is our earnest effort to find solution for this problem. At present, the segregation of the arecanuts is done manually by skilled labour at various collection centres of respective organisations. As described in chapter 1, due to the shortage of skilled labour, the segregation process is delayed. With the availability of AI state-of-art tools, it is now possible to use these technologies more resourcefully for nut segregation and make it farmer friendly. Also, these technologies can reduce the cost of instrumentation development drastically. Keeping these requirements in mind, we plan to develop an AI-based segregation algorithm to classify the arecanuts with an accuracy more than 95%.

2.6.1 The main objectives

After going through the various literature survey and the present technologies used in the arecanut segregation, it was felt that farmer friendly automated segregation unit is the need of the hour. Before we finalised our objectives we also looked at other fruit classification methods as described in section 1.5.1. With all these inputs in our hand and the problems faced by farmers while trading their arecanut crops in market, we have set following objectives.

- Design of an image acquisition system for capturing quality images.
- Creating a complete database of Goan variety of Arecanut.
- Design of Custom Deep Learning (DL) algorithms for accurate segregation.
- Design and Development of an AI based system for segregation.
- Performance evaluation of the system designed.

It was our confidence that we could achieve these goals in a reasonable time limit of 5 to 6 years. We have mentioned in chapter 1 that, there are many problems faced by the farmers in the trade of arecanuts. We have limited our solution to address some of them within the time limit of doctoral dissertation period. Many of the additional solutions for the instrumentation required for addressing the problems are mentioned in the future scope of section 7.6 in the last chapter.

2.7 References

1. Md. M. Rahman, Md. A. Basar, T. S. Shinti, Md. S. I. Khan, H. Md. H. Babu, and K. M. M. Uddin, "A deep CNN approach to detect and classify local fruits through a web interface," *Smart Agricultural Technology*, vol. 5, p. 100321, Oct. 2023, doi: 10.1016/j.atech.2023.100321.
2. K. Albarrak, Y. Gulzar, Y. Hamid, A. Mehmood, and A. B. Soomro, "A Deep Learning-Based Model for Date Fruit Classification," *Sustainability*, vol. 14, no. 10, p. 6339, May 2022, doi: 10.3390/su14106339.
3. V.-N. Pham, Q.-H. Do Ba, D.-A. Tran Le, Q.-M. Nguyen, D. Do Van, and L. Nguyen, "A Low-Cost Deep-Learning-Based System for Grading Cashew Nuts," *Computers*, vol. 13, no. 3, p. 71, Mar. 2024, doi: 10.3390/computers13030071.
4. M. Yurdakul, İ. Atabaş, and Ş. Taşdemir, "Almond (Prunus dulcis) varieties classification with genetic designed lightweight CNN architecture," *Eur Food Res Technol*, May 2024, doi: 10.1007/s00217-024-04562-4.
5. B. Huang *et al.*, "Applications of machine learning in pine nuts classification," *Sci Rep*, vol. 12, no. 1, p. 8799, May 2022, doi: 10.1038/s41598-022-12754-9.
6. S. S. Kumar, An. Sigappi, G. A. S. Thomas, Y. H. Robinson, and S. P. Raja, "Classification and Analysis of Pistachio Species Through Neural Embedding-Based Feature Extraction and Small-Scale Machine Learning Techniques," *Int. J. Image Grap.*, vol. 24, no. 03, p. 2450032, May 2024, doi: 10.1142/S0219467824500323.
7. D. Singh *et al.*, "Classification and Analysis of Pistachio Species with Pre-Trained Deep Learning Models," *Electronics*, vol. 11, no. 7, p. 981, Mar. 2022, doi: 10.3390/electronics11070981.
8. Z. Ünal and H. Aktaş, "Classification of hazelnut kernels with deep learning," *Postharvest Biology and Technology*, vol. 197, p. 112225, Mar. 2023, doi: 10.1016/j.postharvbio.2022.112225.
9. E. Dönmez, S. Kılıçarslan, and A. Diker, "Classification of hazelnut varieties based on bigtransfer deep learning model," *Eur Food Res Technol*, vol. 250, no. 5, pp. 1433–1442, May 2024, doi: 10.1007/s00217-024-04468-1.
10. R. Pillai, N. Sharma, and R. Gupta, "Classification of Pista Species using Fine-Tuned EfficientNetB3 Transfer Learning Model," in *2023 International Conference on Research Methodologies in Knowledge Management, Artificial Intelligence and Telecommunication*

Engineering (RMKMATE), Chennai, India: IEEE, Nov. 2023, pp. 1–5. doi: 10.1109/RMKMATE59243.2023.10369347.

11. K. A. D. Idress, Y. B. Öztekin, O. A. A. Gadalla, and G. P. Baitu, "Classification of Pistachio Varieties Using Pre-trained Architectures and a Proposed Convolutional Neural Network Model," in *15th International Congress on Agricultural Mechanization and Energy in Agriculture*, vol. 458, E. Cavallo, F. Auat Cheein, F. Marinello, K. Saçılık, K. Muthukumarappan, and P. C. Abhilash, Eds., Cham: Springer Nature Switzerland, 2024, pp. 148–163. doi: 10.1007/978-3-031-51579-8_15.
12. H. Aktaş, T. Kızıldeniz, and Z. Ünal, "Classification of pistachios with deep learning and assessing the effect of various datasets on accuracy," *Food Measure*, vol. 16, no. 3, pp. 1983–1996, Jun. 2022, doi: 10.1007/s11694-022-01313-5.
13. A. Sivaranjani and S. Senthilrani, "Computer Vision-Based Cashew Nuts Grading System Using Machine Learning Methods," *J CIRCUIT SYST COMP*, vol. 32, no. 03, p. 2350049, Feb. 2023, doi: 10.1142/S0218126623500494.
14. M. Rahimzadeh and A. Attar, "Detecting and counting pistachios based on deep learning," *Iran J Comput Sci*, vol. 5, no. 1, pp. 69–81, Mar. 2022, doi: 10.1007/s42044-021-00090-6.
15. J. L. Joseph, V. A. Kumar, and S. P. Mathew, "Fruit Classification Using Deep Learning," in *Innovations in Electrical and Electronic Engineering*, vol. 756, S. Mekhilef, M. Favorskaya, R. K. Pandey, and R. N. Shaw, Eds., Singapore: Springer Singapore, 2021, pp. 807–817. doi: 10.1007/978-981-16-0749-3_62.
16. N.-E.-A. Mimma, S. Ahmed, T. Rahman, and R. Khan, "Fruits Classification and Detection Application Using Deep Learning," *Scientific Programming*, vol. 2022, pp. 1–16, Nov. 2022, doi: 10.1155/2022/4194874.
17. Y. Fu, M. Nguyen, and W. Q. Yan, "Grading Methods for Fruit Freshness Based on Deep Learning," *SN COMPUT. SCI.*, vol. 3, no. 4, p. 264, Jul. 2022, doi: 10.1007/s42979-022-01152-7.
18. A. Soleimanipour, M. Azadbakht, and A. Rezaei Asl, "Cultivar identification of pistachio nuts in bulk mode through EfficientNet deep learning model," *Food Measure*, vol. 16, no. 4, pp. 2545–2555, Aug. 2022, doi: 10.1007/s11694-022-01367-5.
19. Y.-D. Zhang *et al.*, "Image based fruit category classification by 13-layer deep convolutional neural network and data augmentation," *Multimed Tools Appl*, vol. 78, no. 3, pp. 3613–3632, Feb. 2019, doi: 10.1007/s11042-017-5243-3.

20. A. Taner, Y. B. Öztekin, and H. Duran, “Performance Analysis of Deep Learning CNN Models for Variety Classification in Hazelnut,” *Sustainability*, vol. 13, no. 12, p. 6527, Jun. 2021, doi: 10.3390/su13126527.
21. Y. Han, Z. Liu, K. Khoshelham, and S. H. Bai, “Quality estimation of nuts using deep learning classification of hyperspectral imagery,” *Computers and Electronics in Agriculture*, vol. 180, p. 105868, Jan. 2021, doi: 10.1016/j.compag.2020.105868.
22. K. G. Panchbhai, M. G. Lanjewar, V. V. Malik, and P. Charanarur, “Small size CNN (CAS-CNN), and modified MobileNetV2 (CAS-MODMOBNET) to identify cashew nut and fruit diseases,” *Multimed Tools Appl*, Apr. 2024, doi: 10.1007/s11042-024-19042-w.
23. V. Gautam, A. Vajpee, and Abhishek, “Transfer learning based optimized deep neural network for pistachio classification,” Stavropol, Russia, 2023, p. 060010. doi: 10.1063/5.0178612.
24. J. L. Rojas-Aranda, J. I. Nunez-Varela, J. C. Cuevas-Tello, and G. Rangel-Ramirez, “Fruit Classification for Retail Stores Using Deep Learning,” in *Pattern Recognition*, vol. 12088, K. M. Figueroa Mora, J. Anzurez Marín, J. Cerda, J. A. Carrasco-Ochoa, J. F. Martínez-Trinidad, and J. A. Olvera-López, Eds., Cham: Springer International Publishing, 2020, pp. 3–13. doi: 10.1007/978-3-030-49076-8_1.
25. L. Lisda, K. Kusrini, and D. Ariatmanto, “Classification of Pistachio Nut Using Convolutional Neural Network,” *Inf. J. Ilm. Bid. Teknol. Inf. dan Komun.*, vol. 8, no. 1, pp. 71–77, Jan. 2023, doi: 10.25139/inform.v8i1.5685.
26. Y. Yu, X. An, J. Lin, S. Li, and Y. Chen, “A vision system based on CNN-LSTM for robotic citrus sorting,” *Information Processing in Agriculture*, vol. 11, no. 1, pp. 14–25, Mar. 2024, doi: 10.1016/j.inpa.2022.06.002.
27. K.-Y. Huang, “Detection and classification of arecanuts with machine vision,” *Computers & Mathematics with Applications*, vol. 64, no. 5, pp. 739–746, Sep. 2012, doi: 10.1016/j.camwa.2011.11.041.
28. S. Siddesha, S. K. Niranjan, and V. N. Manjunath Aradhya, “Texture based classification of arecanut,” in *2015 International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT)*, Davangere: IEEE, Oct. 2015, pp. 688–692. doi: 10.1109/ICATCCT.2015.7456971.
29. Suresha M, Ajit Danti, S. K Narasimhamurthy. Classification of Diseased Arecanut based on Texture Features. National Conference on Recent Advances in Information Technology. NCRAIT, 3 (February 2014), 1-6.

30. A. Danti and Suresha, “Segmentation and Classification of Raw Arecanuts Based on Three Sigma Control Limits,” *Procedia Technology*, vol. 4, pp. 215–219, 2012, doi: 10.1016/j.protcy.2012.05.032.
31. Meghana D R and Prabhudeva S, “Image Processing based Arecanut Diseases Detection Using CNN Model,” *IJARSCT*, pp. 747–752, Jun. 2022, doi: 10.48175/IJARSCT-5154.
32. Dhanuja K C and PES College Of Engineering, “Arecanut Disease Detection using Image Processing Technology,” *IJERT*, vol. V9, no. 08, p. IJERTV9IS080352, Sep. 2020, doi: 10.17577/IJERTV9IS080352.
33. B. N. K. Et. Al., “Classification and Grading of Arecanut Using Texture Based Block-Wise Local Binary Patterns,” *TURCOMAT*, vol. 12, no. 11, pp. 575–586, May 2021, doi: 10.17762/turcomat.v12i11.5931.
34. S. Shedthi B, M. Siddappa, S. Shetty, and V. Shetty, “Classification of arecanut using machine learning techniques,” *IJECE*, vol. 13, no. 2, p. 1914, Apr. 2023, doi: 10.11591/ijece.v13i2.pp1914-1921.
35. F. M. Sarimole and A. Rosiana, “Classification of Maturity Levels in Areca Fruit Based on HSV Image Using the KNN Method,” *JAETS*, vol. 4, no. 1, pp. 64–73, Sep. 2022, doi: 10.37385/jaets.v4i1.951.
36. A. S and A. Hegde, “Detection and classification of arecanut diseases,” in *2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC)*, Coimbatore, India: IEEE, Aug. 2021, pp. 1092–1097. doi: 10.1109/ICESC51422.2021.9532754.
37. A. Hegde, V. S. Sadanand, C. G. Hegde, K. M. Naik, and K. D. Shastri, “Identification and categorization of diseases in arecanut: a machine learning approach,” *IJEECS*, vol. 31, no. 3, p. 1803, Sep. 2023, doi: 10.11591/ijeecs.v31.i3.pp1803-1810.
38. S. Patil, A. Naik, M. Sequeira, G. Naik, and J. Parab, “An Algorithm for Pre-processing of Arecanut for Quality Classification,” in *Second International Conference on Image Processing and Capsule Networks*, vol. 300, J. I.-Z. Chen, J. M. R. S. Tavares, A. M. Iliyasu, and K.-L. Du, Eds., Cham: Springer International Publishing, 2022, pp. 79–93. doi: 10.1007/978-3-030-84760-9_8.
39. S. B. Mallikarjuna *et al.*, “CNN BASED METHOD FOR MULTI-TYPE DISEASED ARECANUT IMAGE CLASSIFICATION,” *MJCS*, vol. 34, no. 3, pp. 255–265, Jul. 2021, doi: 10.22452/mjcs.vol34no3.3.

40. Jyothi K, Sindhu M Hegde, Sumedha K S, Sushma C R, and Thanushree D C, “Grading of Arecanut Using Machine Learning Techniques,” *IJSRCSEIT*, pp. 33–39, Jul. 2022, doi: 10.32628/CSEIT2283119.
41. S. Patil, A. Naik, M. Sequeira, and J. Parab, “A Dehusked Arecanut Classification Algorithm Based on 10-Fold Cross-Validation of Convolutional Neural Network,” in *Advanced Network Technologies and Intelligent Computing*, vol. 1798, I. Woungang, S. K. Dhurandher, K. K. Pattanaik, A. Verma, and P. Verma, Eds., Cham: Springer Nature Switzerland, 2023, pp. 35–45. doi: 10.1007/978-3-031-28183-9_3.
42. S. Patil, A. Naik, and J. Parab, “Efficient Deep Learning model for de-husked Arecanut classification,” *JANS*, vol. 15, no. 4, pp. 1529–1540, Dec. 2023, doi: 10.31018/jans.v15i4.5067.
43. H. Chandrashekhara and M. Suresha, “Classification of Healthy and Diseased Arecanuts using SVM Classifier,” *ijcse*, vol. 7, no. 2, pp. 544–548, Feb. 2019, doi: 10.26438/ijcse/v7i2.544548.
44. S. M and A. Danti, “Construction of Co-occurrence Matrix using Gabor Wavelets for Classification of Arecanuts by Decision Trees,” *IJAIS*, vol. 4, no. 6, pp. 33–39, Dec. 2012, doi: 10.5120/ijais12-450775.

CHAPTER 3

METHODOLOGY AND INSTRUMENTATION

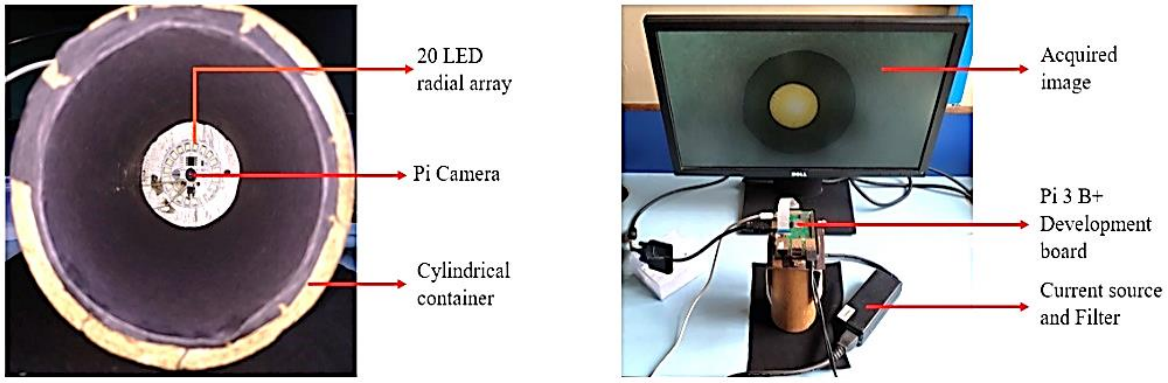
In this chapter, we will discuss the design and development of the image acquisition system, Dataset description, and various image processing techniques utilized in the preprocessing of data before giving it to the various deep learning algorithms.

In our research work, we are dealing with the quality classification of Arecanut from the Konkan belt of India, particularly from the state of Goa. A framework of the proposed Arecanut classification system is depicted in Figure 3.1. The proposed system is divided into five main steps: Image acquisition, Pre-processing, Image segmentation, Classification models and Model performance analysis.

Figure 3.1: Framework of the proposed method

3.1. Image acquisition system

A special setup was designed for the creation of a preliminary database as there was no publicly available database of the Arecanut images. In this setup, a camera is mounted at the top, and a sample table is placed below. The distance between the camera and the sample table is approximately 14 cm. A radial arrangement of 20 white LEDs evenly illuminating the sample is placed surrounding the camera. The sample and the camera are shielded from stray light by using a hollow cylinder pasted with black matt paper on its inner side as shown in figure 2 a. An AC power source of 220V, 50 Hz is applied to the LEDs through a constant current DC converter. Also, a 220 µF/ 450 V capacitor is connected at DC output to reduce flicker in the illumination. In this setup, we have used a low-cost 5MP lightweight Pi camera module, which communicates with the Raspberry Pi 3 B+ board using the MIPI camera serial interface protocol. To reduce the backscattered light, a sample (arecanut) is placed over a black cloth. Figure 3.2, shows the data acquisition setup designed for capturing images of Arecanuts.



(a)Top view of imaging setup (b)Components of imaging setup

Figure 3.2: Data acquisition setup

3.2 Dataset description

In Chapter 1, we have mentioned about seven classes of arecanuts. There is little difference between Supari and Safed on a textural basis. There is always confusion while segregating a Supari and a Safed class, even among the skilled experts. This is because of the micro-miniature cracks developed in the Safed class due to extra dryness. Similarly, it requires a minute observation to separate Laal from Vench. The Laal variety has a hole at the bottom and sometimes may have minor cracks on the surface with a little husk at the bottom or sideways. Similarly, the Vench variety has some cracks and a little husk sideways and may have a hole at the bottom. To differentiate these small changes, we require high-resolution cameras as well as different views of the arecanut, which need additional cameras and high level computational backend support. Thus, it leads to a more complex design of the system, resulting in extended time for design and development and also more financial support. However, if we combine the above-mentioned classes, we can limit our time requirement within a reasonable limit and with available financial resources. The matter was deliberated with the Traders (Goa Bagayatdar, Ponda), who agreed with our observation and suggested that even five-class segregation is a worthwhile solution for quick arecanut segregation.

Therefore, the two classes, Supari and Safed classes, are combined in our classification as Supari. Similarly, Laal and Vench classes were combined under a common category as Laal. Therefore, henceforth, in our work, we use five classes of arecanuts for classification, namely Supari, Laal, Tukada, Kharad, and Baad, as shown in Fig 3.3.

In this work, A dataset of 300 nuts (60 per class) is created using a specially designed setup shown in figure 3.2. Here, we have created a database consisting of five classes, namely

Supari, Laal, Tukada, Kharad, and Baad. The images acquired of all the mentioned categories are shown in Figure 3.3.

Figure 3.3: Various classes of arecanuts

(a) Supari (b) Laal (c) Tukada (d) Kharad (e) Baad

Figure 3.3 (a) is an image of the supari class nut. This class of nut is the costliest among all and has no defects. Figure 3.3(b) shows the second class of nuts called Laal. This nut has a hole at the bottom. This class nut is priced marginally below Supari class. Figure 3.3(c) is a picture of Tukada. These nuts are a broken variety of Supari and Laal class and are accordingly priced below them. The Tukada variety is mostly generated when nuts are peeled using an automatic de-husking machine. Figure 3.3(d) shows the Kharad class nut. This variety has a husk on the major portion of the nut. This husk remains on the nut due to improper drying of the nut and is mostly available in nuts that are peeled using an automatic peeling/ de-husking machine. Since the processing of this class of nut in the industry requires additional steps, this class is priced below Tukada. The Baad class shown in Figure 3.3(e) is the lowest quality of nuts and is priced the lowest. This nut has less weight, is mostly uneven in shape, and is porous in nature.

3.3 Image Pre-Processing

Image pre-processing is crucial in various image analysis tasks, especially in scientific, industrial, and ML/DL applications [1]. It prepares raw images for better analysis, enhancing accuracy and efficiency in downstream processes. The key benefits of image pre-processing include [2]:

Noise Reduction: Raw images often contain noise due to environmental factors, sensor imperfections, or low-quality equipment. Pre-processing removes this noise, leading to more explicit images and more accurate results in subsequent analysis.

Improved Image Quality: Techniques like contrast enhancement and sharpening are applied to improve the visibility of important features, enabling better identification and analysis of critical regions.

Consistent Image Dimensions: Resizing and normalization ensure that all images used in a dataset have uniform dimensions and pixel ranges, making them suitable for automated processing and ML/DL models.

Feature Enhancement: Pre-processing helps enhance specific features such as edges, textures, or patterns in the image. This is crucial for accurate object detection, classification, or segmentation tasks.

Reduction of Computational Complexity: By converting images to grayscale, resizing, or reducing irrelevant information (e.g., background removal), pre-processing reduces the size and complexity of the data, optimizing processing speed and resource usage.

Standardization: Pre-processing ensures that images are standardized across different acquisition conditions, making comparing and analyzing datasets collected in varying lighting, angles, or sensor settings easier.

Segmentation for Object Detection: Pre-processing techniques like thresholding and edge detection allow for the separation of objects from the background, aiding in object recognition, tracking, or measurement.

Facilitating ML and AI Models: Properly pre-processed images improve the performance of ML models, increasing the accuracy of predictions. Models trained on high-quality, noise-free images are less likely to overfit and more likely to generalize well.

Augmentation of Data: By applying transformations (like rotation, flipping, or zooming), pre-processing artificially increases the dataset size, which improves the robustness and generalization of ML/DL algorithms.

Handling Variability: Pre-processing helps mitigate the effects of variability caused by image capture conditions, making the dataset more uniform and thus improving the reliability of analysis.

Image pre-processing is essential for optimizing data quality, reducing complexity, and improving the performance of image analysis systems, particularly in ML and computer vision

tasks. Enhancing features, standardizing input, and removing noise enables more accurate and efficient processing of images.

3.3.1 Contrast Enhancement in Image Processing

Contrast in an image refers to the difference in luminance or colour that makes an object distinguishable from others in the same image [3]. Specifically, it represents the difference between an image's lightest and darkest parts. Enhancing contrast involves adjusting the image to improve the visual distinction between different regions or features, making important structures more visible.

Methods of Contrast Enhancement: Contrast enhancement is a key image processing technique used to improve the visibility and distinction between different features in an image. It adjusts the intensity or colour distribution, making important details clearer for analysis [4].

Below are some commonly used methods:

1. **Linear Methods**
    i. **Contrast Stretching**: This method stretches the range of intensity values in the image. The minimum and maximum pixel values are scaled to the total intensity range (e.g., 0 to 255 for an 8-bit image). As a result, areas that were once difficult to differentiate due to similar brightness levels become more distinct.
    ii. **Histogram Equalization**: This technique redistributes the pixel intensity distribution of an image, enhancing the contrast, especially in images where the pixel intensities are concentrated within a narrow range. By spreading out the most frequent intensity values, it improves the contrast across the entire image.
2. **Non-Linear Methods**
    i. **Gamma Correction**: This adjusts the image's luminance by applying a non-linear transformation to each pixel. The gamma value controls the image's brightness; gamma values less than 1 darken the image, while values greater than 1 lighten it. This technique selectively enhances contrast in either bright or dark regions.
    ii. **Logarithmic Transformation**: Logarithmic operations can be applied to compress the dynamic range of an image. This method is particularly effective in enhancing contrast in darker regions of the image, making subtle details more visible.
    iii. **Adaptive Histogram Equalization (AHE)**: Unlike standard histogram equalization, AHE improves local contrast. It divides the image into smaller

regions and applies histogram equalization to each, which enhances contrast in areas with varying brightness levels.

Importance of Contrast Enhancement: Contrast enhancement is crucial in image processing as it improves the visibility of details in an image by increasing the difference between light and dark areas [5,6]. This technique is especially important in various fields for the following reasons:

Improved Visibility of Features: By increasing contrast, subtle differences between regions of an image become more prominent, making important details easier to detect. This is especially useful in applications like medical imaging, where identifying small features is critical.

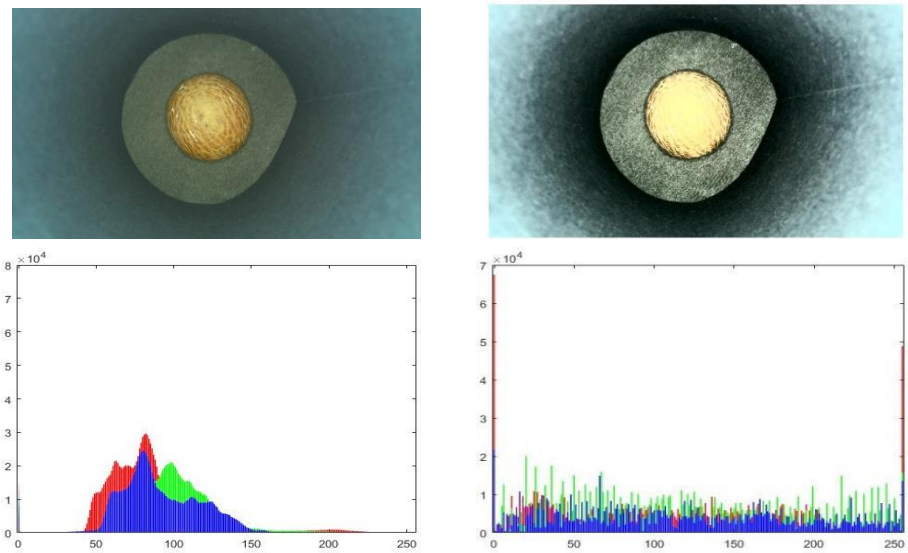
Better Edge Detection: Higher contrast sharpens the boundaries between objects, aiding edge detection algorithms and improving object recognition tasks.

Improved Aesthetics: For visual media, contrast enhancement enhances image quality by making it more appealing and more accessible to interpret, thus improving user experience.

Enhanced Image Analysis: In scientific and industrial imaging, enhanced contrast is essential for accurate image analysis, allowing better segmentation, classification, and pattern recognition.

Contrast enhancement manipulates pixel intensity to improve visual differentiation between different parts of an image. Whether a linear method, like contrast stretching and histogram equalization, or non-linear techniques, like gamma correction and adaptive histogram equalization, the goal is to make details more transparent and enhance the interpretability of the image for various applications.

The contrast of an image is the difference between the reflected luminance between two surfaces. Thus, the contrast of an image can be enhanced by manipulating the pixels in the spatial domain using linear or nonlinear methods. This manipulation of pixels results in an improved visual distinction between various image structures, thus increasing the contrast.



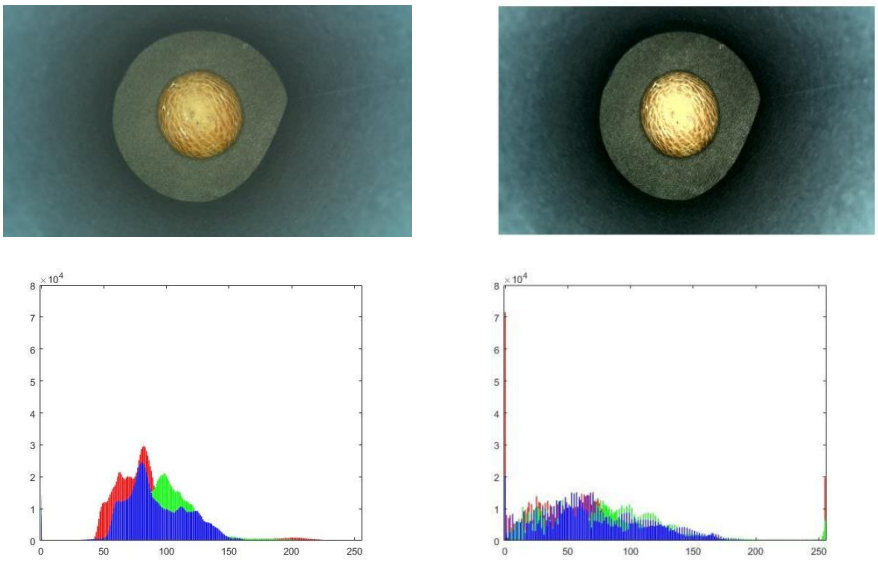
(a) Original image **(b) Histogram equalized image**

(c) Histogram of original image **(d) Histogram of histogram equalized image**

Fig. 3.4: Histogram Equalization

Histogram Equalization: Histogram of an image is the graphical representation of the frequency of repetition of pixel intensity values. Histogram Equalization is an image-processing algorithm that increases the contrast of the image by stretching out the histogram of the original image, i.e., the most frequent intensity values are spread out. This allows the areas of lower contrast to be converted into higher contrast [7, 8]. Figure 3.4 illustrates the histogram equalized images and their corresponding histograms.

Intensity Adjustment: Intensity Adjustment is a contrast enhancement technique, where, the intensity map of the original image is mapped to a new range of intensity. This allows to brighten or darken the intensity values of the original image to the higher or lower range of [0 255] intensity map. The contrast of the image can be increased by mapping the intensity values of the original image to fill the entire [0 255] intensity range. Figure 3.5 illustrates the intensity adjusted images and the corresponding histograms are also displayed [9, 10].



(a) Original image **(b) Intensity adjustment of original image**

(c) Histogram of original image **(d) Histogram of intensity adjusted image**

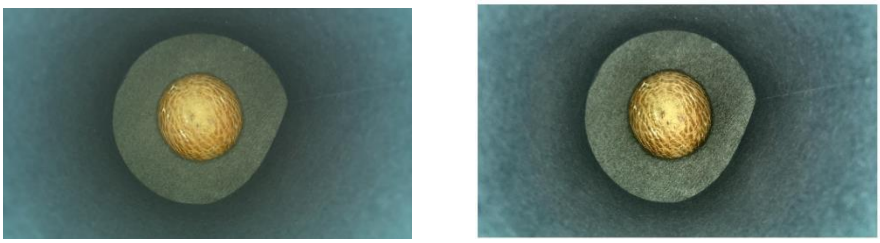
Fig. 3.5: Intensity adjustment and corresponding Histograms

Contrast-limited Adaptive Histogram Equalization (CLAHE): CLAHE is an image processing algorithm, that is used to enhance the contrast of the image. CLAHE algo- rithm operates on small, non-overlapping regions of the image, rather than the whole image like Histogram Equalization. These small regions are called tiles. A histogram based contrast enhancement is performed on each tile. The histograms obtained are then clipped to a level defined by the clip limit and then redistributed. This prevents the over enhancement of homogeneous regions. The clip limit is obtained using Equation 1. Then, using bilinear interpolation, the neighboring tiles are combined to avoid artificially induced boundaries [11, 12]. It may be seen from Figure 3.6 that, CLAHE technique gives a sharp enhancement in the edges.

$$clip = A = \frac{N}{G} + \alpha\left(N - \frac{N}{L}\right) \text{----------(1)}$$

Where:

N: is the no. of pixels in each tile
G: Total no. of grey levels in the input image I(x,y)
α: is the clip factor



(a) Original image **(b) CLAHE equalized image**

Fig. 3.6: Contrast-limited adaptive histogram equalization

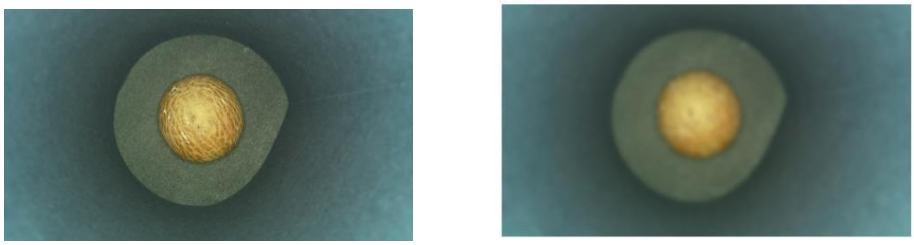
3.3.2 Image Filtering

Image filtering algorithm aims to reduce the noise, image smoothening, image enhancement and edge detection. It is a neighbourhood process in which a kernel of a fixed size is convolved with a pixel and its neighbours. In this section, we discuss Gaussian Filter, Anisotropic Diffusion Filter, and Median Filter.

Gaussian Filter: A Gaussian Filter, also known as a smoothing or blurring filter, is a low-pass linear filter. The weights of this filter are chosen in accordance with the shape of the Gaussian function. In the case of image processing, we utilize a 2-dimensional Gaussian function given by Equation 2. This filter is often used to remove Gaussian noise and other noises such as salt and pepper. As it is a low-pass filter, it removes high-frequency components such as edges and fine details [13, 14]. The steps for Gaussian filtering are shown in Figure 3.7.

$$G(x,y)\frac{1}{2\pi\sigma^2}e^{-\frac{x^2+x^2}{2\sigma^2}}$$

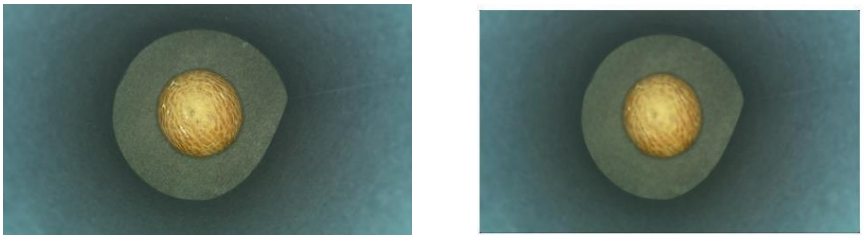
Where σ: is the standard deviation



(a) Original image **(b) Gaussian Filtering of image**

Fig. 3.7: Gaussian Filtering

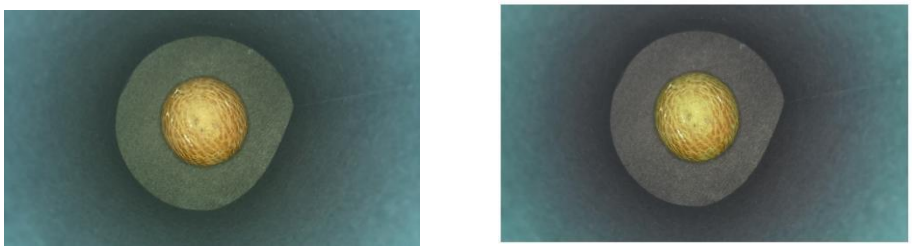
Median Filtering: Median Filter is a nonlinear image filter that is very useful in removing random and impulse noise, such as salt and pepper noise, while still being sensitive to edge information. The median filtering operation is achieved by making use of a sliding window. This window replaces the value at the window center with the median values of the neighboring pixels in the window [14, 15, 16]. The Figure3.8 illustrates the images using Median Filtering.



(a) Original image **(b) Median Filtering of image**

Fig.3.8: Median Filtering

Anisotropic Diffusion Edge Preserving Filter: Anisotropic Diffusion filter is a non linear filter, also known as Perona-Mallik Filter, that is used to filter image noise, while at the same time aiming to preserve image details such as edges, lines and other image structures. The filter is based on the Partial Differential Equation of heat equation or diffusion equation. The diffusion coefficient is chosen such that there is maximal intra-region smoothing while respecting the edge information [17, 18]. The images using Anisotropic Diffusion filtering are shown in Figure 3.9.



(a) Original image **(b) Anisotropic Filtering of image**

Fig.3.9: Anisotropic Filtering

3.3.3 Image Segmentation

Image segmentation is a process by which an image is partitioned into a pixel groups called segments, that is represented by a mask or a label. This segmentation thus reduces the complexity of an image, thereby allowing simplification of the analysis algorithm. In this section, we explore K-means Segmentation and Canny Edge Detector for Image Segmentation.

K-Means Image Segmentation: K-Means Image Segmentation is an unsupervised ma- chine learning algorithm. It is a distance based or centroid based algorithm, wherein, an image is segmented into *k* segments or clusters. Each data point or pixel is assigned to a cluster based on its mean from the centroid, and thus their similarity to each other. The Euclidean distance between each pixel and the centroid is given by Equation 3.

$$d = \|I(x,y) - C_k\| \text{----------(3)}$$

Where:

$I(x,y)$: is the input image

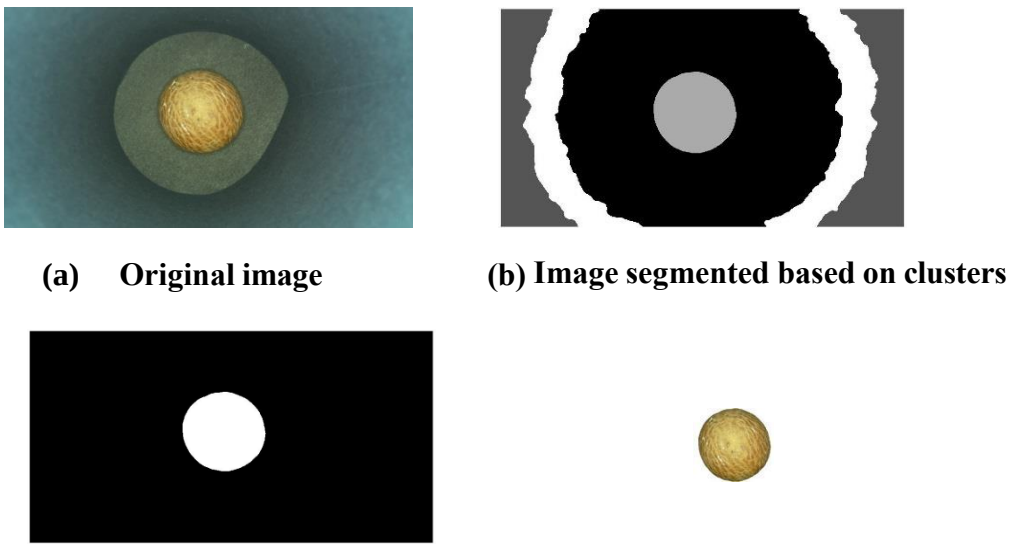
c_k: are cluster centers of k clusters

d: is the Euclidean distance

Upon determination of the Euclidean distance d, the pixel is assigned to a cluster with which it has the least distance. Further, the new position of the centers is then calculated iteratively using the equation given in Equation 4.

$$C_k = \frac{1}{k} \sum_{x \in C_k} \sum_{y \in C_k} I(x,y)$$

The steps in image segmentation using K-Means segmentation are illustrated in Figure 3.10 other [19, 20].



(a) Original image **(b) Image segmented based on clusters**

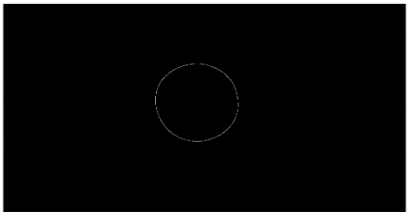
(c) Cluster with the Arecanut image segment (d) Segmented Arecanut image

Fig. 3.10: Stages with K-Means Clustering Segmentation of the Areca nut image.

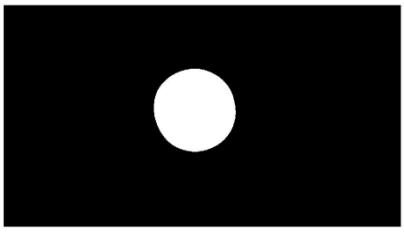
3.3.4 Canny Edge Detection (CED)

Canny Edge detector is a multistage algorithm that performs edge detection, while still performing noise suppression. The CED consists of an initial Noise Reduction stage using the Gaussian filter. The Noise Reduction stage is then followed by determining the intensity gradient of the image in the horizontal and vertical direction. These detected edges are thinned out to 1-pixel width by performing Non-Maximum Suppression, where pixels that do not constitute an edge are removed. A Double Thresholding is then applied, which suppresses gradient information below a minimum value *minVal*, and gradient value above a maximum value *maxVal* are marked as contributing to a sure edge. The information lying between the range of [*minValmaxVal*] is then tracked using Hysteresis Edge tracking. The gradient information lying in [*minValmaxVal*] is converted into a confirmed edge, only if an immediately connected neighboring edge is also a confirmed edge [21, 22]. The steps in image segmentation using Canny Edge Detector are illustrated in Figure 3.11.

(a) Original image



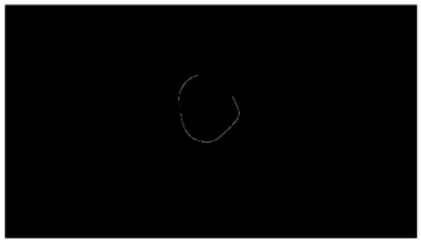
(b) Canny Edge of Areca nut detected



(c) Generation of a mask from the detected edge (d) Canny Edge of Areca nut detected

Fig. 3.11: Stages with Canny Edge Detection Segmentation of the Arecanut image.

In the image segmentation algorithms discussed, K-Means segmentation was able to segment all the images in the database efficiently. However, the Canny Edge Detector could not be used to detect the edges automatically for all the images in the database. This is because certain images need threshold values that are specific to that image. Thus, it will impede the process of segmentation in real-time applications. The failure to completely detect the edge of the Areca nut is illustrated in Figure 3.12.

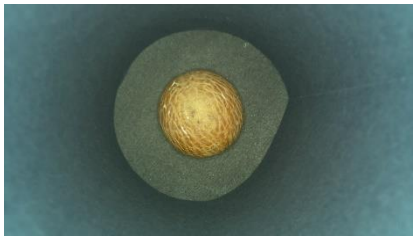


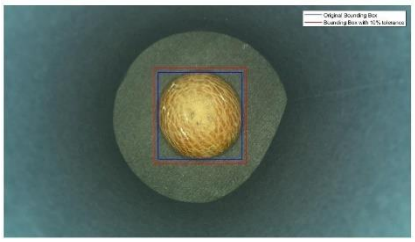
(a) Original image **(b) Canny Edge of Areca nut detected**

Fig. 3.12: Canny Edge Detection failure in a sample image from the database

3.3.5 Image Cropping

The image, after being operated on the above possible pre-processing stages, will be cropped before being presented to a classification algorithm. In order to maintain a standard size for all the images in the database, Arecanut, with the largest size, is detected from the entire database. We then determine a standard size of the Region of Interest, which is calculated using a bounding box for the largest Areca nut. The final bounding box is increased in size by 10%, to accommodate Areca nuts of extraordinary sizes in real-time application. The images at the different stages of the image cropping are shown in Figure 3.13.





(a) **Image of largest Areca nut** (**b) Detection of bounding box with 10% tolerance**

(c) Final cropped image with 10% additional dimension

Fig. 3.13: Various steps in cropping Areca nut images.

3.4 Results of Pre-processing

Here we analyze the results of various preprocessing stages. All eight different techniques for pre-processing Arecanut images for the purpose of their segregation into various classes, such as Supari, Laal, Kharad, Tukada, and Baad, have been discussed. Primarily, we have used three algorithms, namely, Contrast Enhancement, Image Filtering, and Image Segmentation. In Contrast Enhancement algorithms, from Figures 3.4, 3.5, and 3.6, we observed that CLAHE gives the best image contrast without deteriorating the image and maintaining good texture details. In Image Filtering algorithms, from Figures 3.7, 3.8, and 3.9, Anisotropic Diffusion Edge Preserving Filter provides the best edge and texture preservation while it gives high-quality filtering and smoothing of the images.

3.5 References

1. R. Archana and P. S. E. Jeevaraj, “Deep learning models for digital image processing: a review,” *Artif Intell Rev*, vol. 57, no. 1, p. 11, Jan. 2024, doi: 10.1007/s10462-023-10631-z.
2. M. Sonka, V. Hlavac, and R. Boyle, “Image pre-processing,” in *Image Processing, Analysis and Machine Vision*, M. Sonka, V. Hlavac, and R. Boyle, Eds., Boston, MA: Springer US, 1993, pp. 56–111. doi: 10.1007/978-1-4899-3216-7_4.
3. C. Ware, “Color,” in *Visual Thinking for Information Design*, Elsevier, 2022, pp. 65–86. doi: 10.1016/B978-0-12-823567-6.00004-5.
4. Ware, Colin. “Color.” In *Visual Thinking for Information Design*, 65–86. Elsevier, 2022. https://doi.org/10.1016/B978-0-12-823567-6.00004-5.
5. Saiwa, “what is image contrast enhancement?,” Accessed: Sep. 21, 2024. Available: https://medium.com/@saiwadotai/what-is-image-contrast-enhancement-46992b75d2
6. E. K. Lee, E. J. Lee, S. Kim, and Y. S. Lee, “Importance of Contrast-Enhanced Fluid-Attenuated Inversion Recovery Magnetic Resonance Imaging in Various Intracranial Pathologic Conditions,” *Korean J Radiol*, vol. 17, no. 1, p. 127, 2016
7. A. C. Bovik, “Basic Gray Level Image Processing,” in *The Essential Guide to Image Processing*, Elsevier, 2009, pp. 43–68. doi: 10.1016/B978-0-12-374457-9.00003-2.
8. A. Kumar R., V. S. Rajpurohit, and B. J. Jirage, “Pomegranate Fruit Quality Assessment Using Machine Intelligence and Wavelet Features,” *Journal of Horticultural Research*, vol. 26, no. 1, pp. 53–60, Jun. 2018, doi: 10.2478/johr-2018-0006.
9. R. Kaur and S. Kaur, “Comparison of contrast enhancement techniques for medical image,” in *2016 Conference on Emerging Devices and Smart Systems (ICEDSS)*, Namakkal, India: IEEE, Mar. 2016, pp. 155–159. doi: 10.1109/ICEDSS.2016.7587782.
10. M. A. Khan *et al.*, “CCDF: Automatic system for segmentation and recognition of fruit crops diseases based on correlation coefficient and deep CNN features,” *Computers and Electronics in Agriculture*, vol. 155, pp. 220–236, Dec. 2018, doi: 10.1016/j.compag.2018.10.013.
11. K. Zuiderveld, “Contrast Limited Adaptive Histogram Equalization,” in *Graphics Gems*, Elsevier, 1994, pp. 474–485. doi: 10.1016/B978-0-12-336156-1.50061-6.
12. S. Aboshosha, O. Zahran, M. I. Dessouky, and F. E. Abd El-Samie, “Resolution and quality enhancement of images using interpolation and contrast limited adaptive histogram equalization,” *Multimed Tools Appl*, vol. 78, no. 13, pp. 18751–18786, Jul. 2019
13. E. S. Gedraite and M. Hadad, “Investigation on the effect of a Gaussian Blur in image

filtering and segmentation," *Proceedings ELMAR-2011*, Oct. 2011, Accessed: Oct. 22, 2024. [Online]. Available: https://www.semanticscholar.org/paper/Investigation-on-the-effect-of-a-Gaussian-Blur-in-Gedraite-Hadad/6c1144d8705840e075739393a10235fcc4cd0f4b

14. M. Zeeshan, A. Prabhu, A. C, and N. S. Rani, "Fruit Classification System Using Multiclass Support Vector Machine Classifier," in *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, Coimbatore, India: IEEE, Jul. 2020, pp. 289–294. doi: 10.1109/ICESC48915.2020.9155817.
15. M. Ohki, M. E. Zervakis, and A. N. Venetsanopoulos, "3-D Digital Filters," in *Control and Dynamic Systems*, vol. 69, Elsevier, 1995, pp. 49–88. doi: 10.1016/S0090-5267(05)80038-6.
16. R. Dinesh and N. K. Bharadwaj, "Possible approaches to arecanut sorting / grading using computer vision: A brief review," in *2017 International Conference on Computing, Communication and Automation (ICCCA)*, Greater Noida: IEEE, May 2017, pp. 1007–1014. doi: 10.1109/CCAA.2017.8229971.
17. P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, no. 7, pp. 629–639, Jul. 1990, doi: 10.1109/34.56205.
18. Computer Science and Engineering discipline, Khulna University, Khulna, Bangladesh, R. Khan, and R. Debnath, "Multi Class Fruit Classification Using Efficient Object Detection and Recognition Techniques," *IJIGSP*, vol. 11, no. 8, pp. 1–18, Aug. 2019, doi: 10.5815/ijigsp.2019.08.01.
19. N. Dhanachandra, K. Manglem, and Y. J. Chanu, "Image Segmentation Using K -means Clustering Algorithm and Subtractive Clustering Algorithm," *Procedia Computer Science*, vol. 54, pp. 764–771, 2015, doi: 10.1016/j.procs.2015.06.090.
20. B. S. Shedthi, S. Shetty, and M. Siddappa, "Implementation and comparison of K-means and fuzzy C-means algorithms for agricultural data," in *2017 International Conference on Inventive Communication and Computational Technologies (ICICCT)*, Coimbatore, India: IEEE, Mar. 2017, pp. 105–108. doi: 10.1109/ICICCT.2017.7975168.
21. J. Canny, "A Computational Approach to Edge Detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986, doi: 10.1109/TPAMI.1986.4767851.
22. N. M. Hussain Hassan and A. A. Nashat, "New effective techniques for automatic detection and classification of external olive fruits defects based on image processing techniques," *Multidim Syst Sign Process*, vol. 30, no. 2, pp. 571–589, Apr. 2019, doi: 10.1007/s11045-018-0573-5.

CHAPTER 4

CLASSIFICATION ALGORITHMS

There are several algorithms in ML techniques for classifying 2D images. Some of them are computationally intensive, while some are fast without compromising accuracy. In this chapter, we have described six different deep-learning algorithms that we found very suitable for arecanut images. The algorithms used in this research work are described below. Deep Learning algorithms are more suited for our application as we are mostly dealing with high resolution 2D images of arecanut. ML has the capacity to achieve heightened prediction accuracy. Various DL algorithms find application in prediction and classification tasks, contributing to diverse fields, including disease detection, bioinformatics, and computer vision. DL encompasses a spectrum of learning types, such as supervised, unsupervised, and reinforcement learning. Continuous global medical and scientific research refines arecanut detection techniques, benefitting from the advantages presented by DL.

4.1 Deep Learning Basics

Deep learning is a subset of ML, which is a subfield of AI. It focuses on algorithms inspired by the brain's structure and function, particularly artificial neural networks. These networks are intended to learn from vast volumes of data by automatically extracting key features and patterns without operator intervention [1].

Deep learning models, specifically deep neural networks (DNNs), are made up of several layers of artificial neurons. Each layer takes inputs from the previous layer and forwards the findings to the next layer, gradually learning sophisticated representations of data. The "deep" in deep learning refers to the model's numerous layers, which enable it to capture nuanced patterns in data.

4.1.1 Important points concerning deep learning:

Neural Networks: Deep learning employs multi-layer neural networks to solve issues, such as convolutional neural networks (CNNs) for image identification and recurrent neural networks (RNNs) for sequence data like text and time series.

Data and Computation: Deep learning requires a lot of labelled data and a lot of computer power (e.g., GPUs) to train.

Applications: Deep learning is used for various tasks, including image classification, speech recognition, natural language processing (NLP), self-driving cars, and medical diagnostics.

4.2 Convolutional Neural Network (CNN)

In recent literature, CNN has received much attention [2, 3]. CNN is a subset of deep learning (DL) algorithms that is particularly effective in classifying data by identifying patterns in images. A CNN is a feed-forward network comprising fundamental components, including a convolutional layer, pooling layer, and activation layer, that are layered together in various ways. The CNN's feature extraction section comprises a variable configuration of convolutional layers, pooling layers, and activation layers [4]. A Fully Connected (FC) layer and the classification layer are then given the extracted characteristics [4]. Each layer type has a particular function. Figure 4.1 shows the customized CNN architecture for areca nut classification and various layers used are explained below.

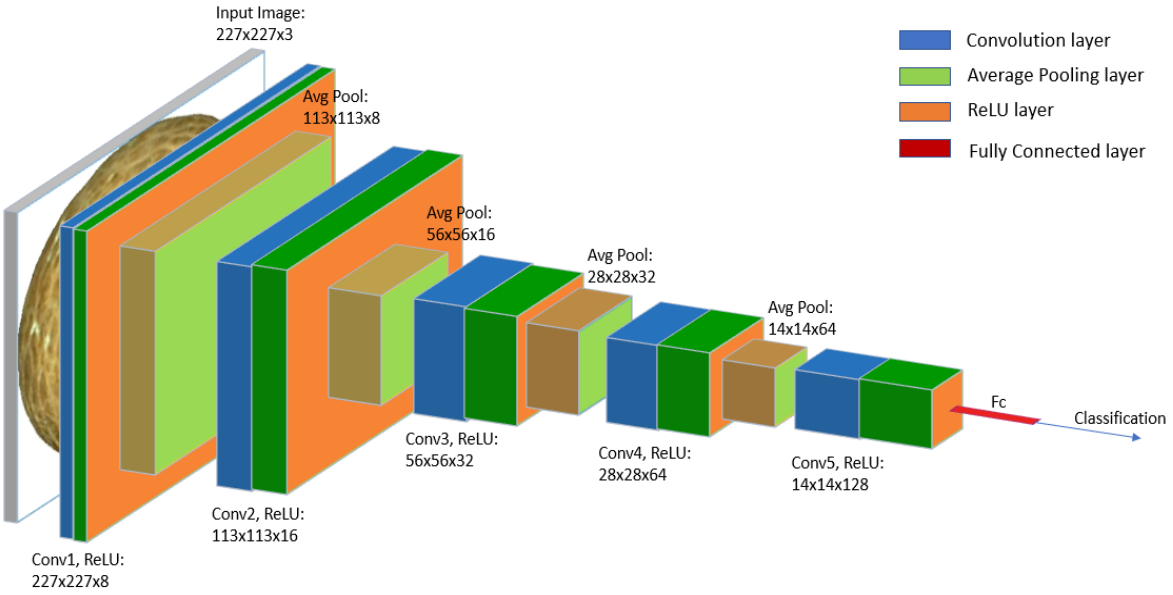


Figure 4.1: The customized CNN model

4.2.1 Convolution layer

A convolution layer is an essential layer of CNN architecture. It is made up of several kernels or filters. These filters provide an N-dimensional metrics-based function map for the input image [5,6].

4.2.2 Activation Layer (ReLU)

The activation ReLU layer helps to introduce non-linearity into the network. It takes the weighted sum of all the inputs from the previous layer and then applies an activation function. It converts the absolute values of the input to positive numbers [5].

4.2.3 Pooling Layer

This layer takes a region of the input feature map and replaces it with the average value of all the pixels in the region. This reduces the size of the feature map, making the model more efficient and reducing the number of parameters used, thereby avoiding the overfitting image [5, 6].

4.2.4 Fully Connected Layer (FC)

In the FC layer, all neurons in the preceding layer are connected to neurons in the subsequent layer. This layer is used in a network's output layers for classification [5].

4.3 AlexNet

AlexNet is a deep convolutional neural network developed by Alex Krizhevsky, Geoffrey Hinton, and Ilya Sutskever. AlexNet uses ReLU as CNN's activation function, and it has been demonstrated that it performs better in deep networks than Sigmoid in terms of gradient dispersion. It consists of eight layers; five convolutional layers, two fully connected layers, and one output layer. To reduce overfitting in the FC layers, AlexNet employed a regularization method called "dropout," which is very effective and is widely used in many DL applications [7]. The AlexNet architecture is given in Figure 4.2. Here, we have used a pre-trained model with a customized output layer, as shown in Fig. 4.2, and details of the modification are given in Appendix-1.

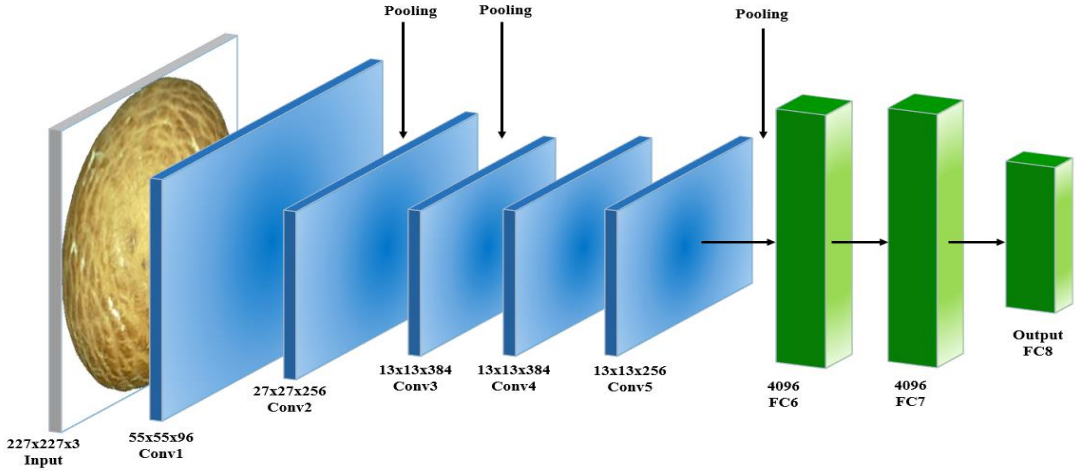


Figure 4.2: The standard AlexNet model architecture

4.4 GoogLeNet

Szegedy et al. [8] suggested GoogLeNet, which demonstrated that it is possible to enhance neural networks for computer vision by approximating the expected optimal sparse structure using easily accessible dense building blocks. GoogLeNet builds a network-in-network using the most progressive Inception network architecture. Then, it uses the convolutional construction blocks to find the optimal design and spatially replicate it [9]. The network was developed with practicality and computational efficiency in mind, allowing inference to be performed on individual devices, even ones with modest memory sizes and limited processing power [8]. When compared to shallower and narrower topologies, the use of GoogLeNet improves the quality gain at a moderate increase in computing needs. GoogLeNet, the 2014 ImageNet competition winner, was recognized as an additional comprehensive and expansive CNN model and showed success with a top-5 error rate of 6.67% [10]. The GoogLeNet architecture is given in Figure 4.3. The detailed code for this model is given at Appendix-2.

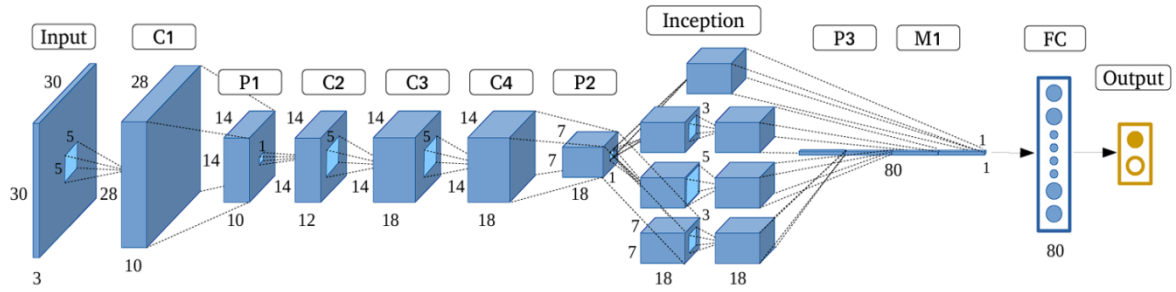


Figure 4.3: The standard GoogLeNet model architecture

4.5 MobileNet model

MobileNet was employed to satisfy the applications involving restricted resources such as low power and low latency models [11,12]. MobileNet comprises a depth-separable convolution that allows a regular convolution into a depth-wise convolution and a pointwise convolution (1 × 1). In the deep convolution layer of each input channel in the MobileNet architecture, a single filter is used. In pointwise convolution, the outputs are combined by a convolution size of 1 × 1 into a penetration-wise convolution. Inputs joint with newly generated outputs in one step are attached with regular convolution filters. The MobileNet architecture is given in Figure 4.4. The detailed code for this model is given at Appendix-3.

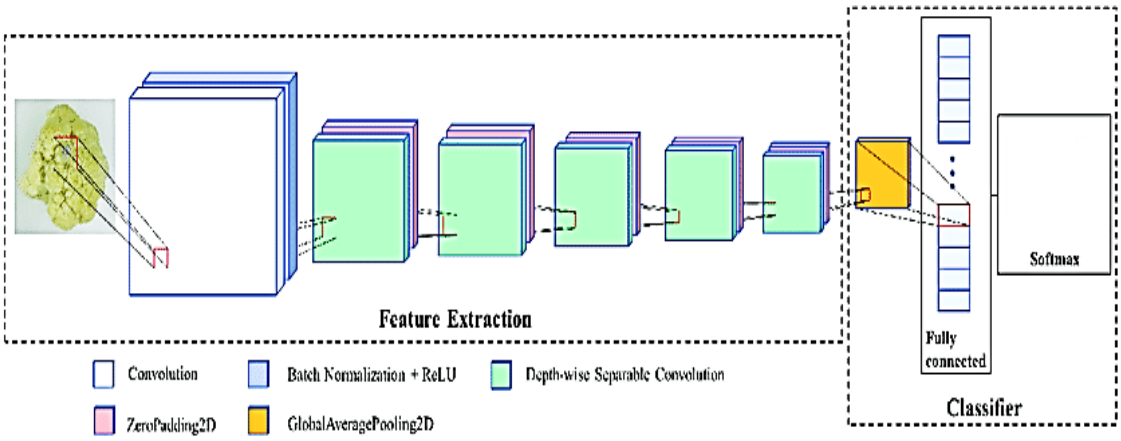


Figure 4.4: The standard MobileNet model architecture

4.6 NasNetMobile

Zoph et al. [13] presented a DCNN design with convolutional, normal, and reduction cells. The Neural Architecture search technique implemented in the CIFAR-10 dataset for finding neural networks led to the creation of NasNet architecture. Reinforcement learning is used by the Neural Architecture Search architecture to determine the optimal hyperparameter, layer type, and layer count. A new architecture was developed by applying resultant architecture to the ImageNet dataset as well. The NasNetMobile design comprises 5.3 million parameters with a 224 × 224 input size [14]. The NasNetMobile architecture is given in Figure 4.5. The detailed code for this model is given at Appendix-4.

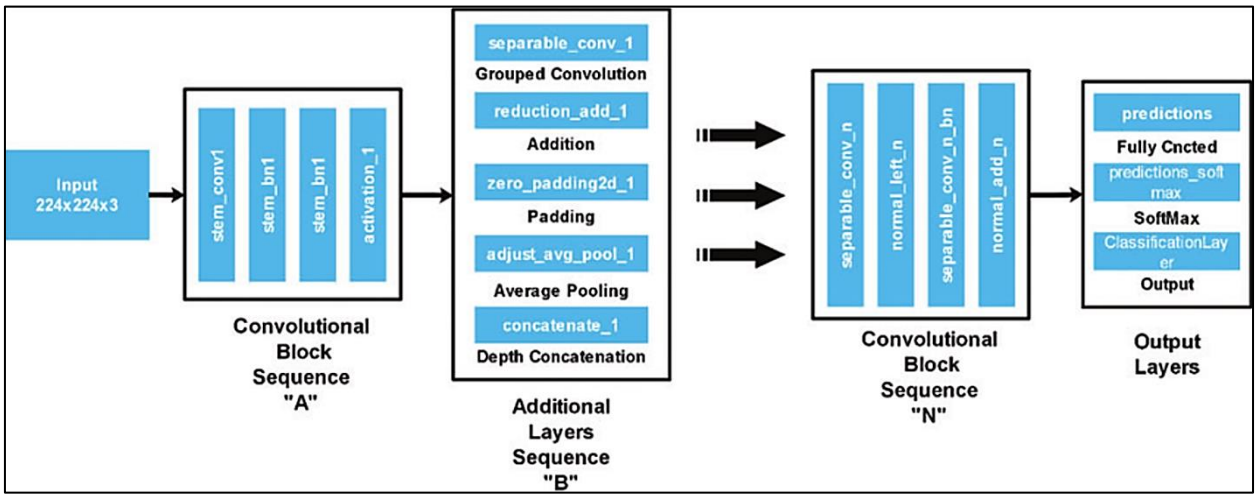


Figure 4.5: The standard NasNetMobile model architecture

4.7 SqeezeNet

Iandola et al. [15] proposed a small CNN architecture, which is a compact version of AlexNet named SqeezeNet, which uses 50 times fewer parameters to achieve AlexNet-level accuracy on ImageNet [16]. Furthermore, it can be compressed to less than 0.5MB using model compression approaches. It can be used with FPGAs and other low-memory hardware. Using the ImageNet Dataset, SqueezeNet has been pre-trained for image classification [17]. It attained a top-5 ImageNet accuracy of 80.3% [15]. The SqeezeNet architecture is given in Figure 4.6. The detailed code for this model is given at Apedix-5.

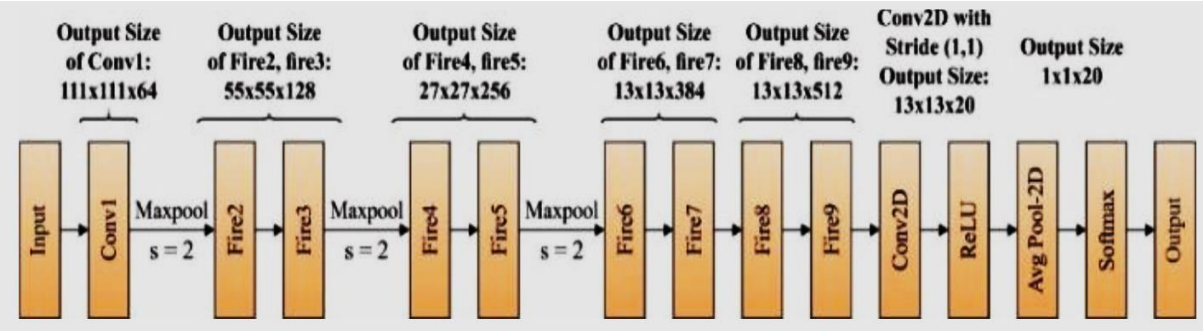


Figure 4.6: The standard SqueezeNet model architecture

4.8 CNN-LSTM

In recent scientific literature, Convolutional Neural Networks (CNN) have garnered substantial attention [2,3]. CNN belongs to the realm of deep learning (DL) algorithms and is particularly adept at categorizing data by discerning intricate patterns within images. CNN aims to learn spatial hierarchies of features automatically and adaptively by backpropagation

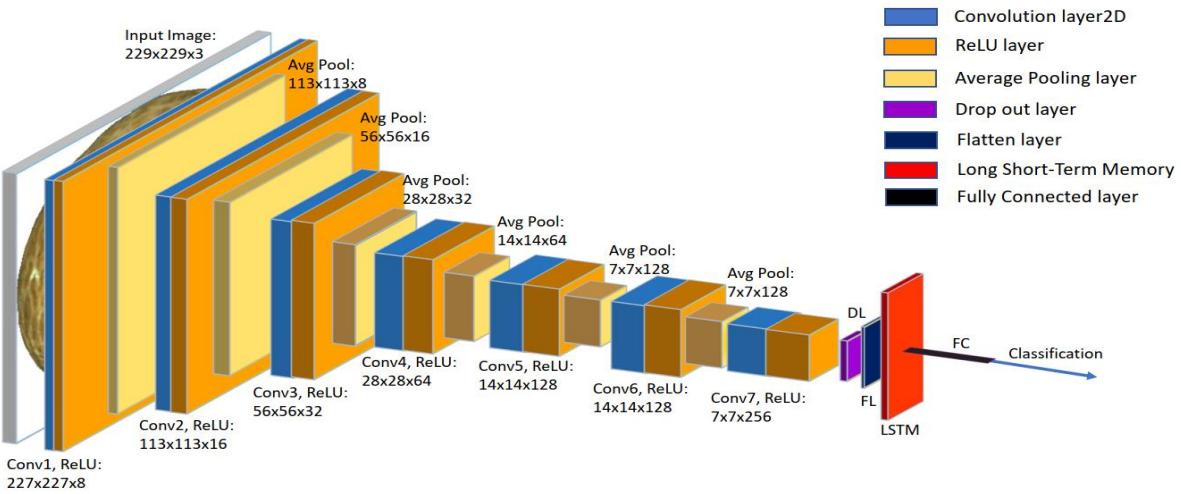


Fig 4.7 Custom CNN-LSTM model

by utilizing several building blocks such as convolution layers, pooling layers, and fully connected layers [6]. The segment of CNN responsible for extracting distinctive features consists of a flexible arrangement of convolutional layers, pooling layers, and activation layers. Subsequently, the extracted characteristics are channelled into a Fully Connected (FC) layer and a classification layer. Recently, LSTM techniques based on the MOT challenge and KITTI benchmarks have been embraced for accurate multi-object tracking [18]. In the proposed method we have provided the custom CNN network output to the input of the LSTM. This allows the LSTM to learn features from the input data that have been learned by the CNN. Fig 4.7 is the custom CNN-LSTM model used for classification. The detailed code for this model is given at Appendix-6.

To evaluate our CNN, we use 10-fold cross-validation. In 10-fold cross-validation, the database is split into 10 distinct folds, of which 9 folds will be used in training, and the 10th fold is used for testing. This means that every sample which was used for testing is now included in the training set and one from the training set is used for testing. Thus, the procedure is repeated 10 times, with each iteration having a new fold from one of the 10 folds for testing [7].

4.9 Custom CNN-LSTM Versus Pre-Trained Models

Here, the custom CNN-LSTM model is compared with the four pre-trained DCNN models. A 5-fold method is applied to the data to get better accuracy. Fig 4.8 displays the overview of the proposed method for the identification of five classes of nuts described in Chapter 3. First, the dataset of various types of Arecanuts is created by capturing their images using an imaging setup. The images need to be resized as per the DL model used. The dataset is split into training and validation data with a ratio of 80:20. A 5-fold method is applied to the data to get better accuracy. The AlexNet, GoogleNet, SqeezeNet, and NasNetMobile pre-trained transfer learning (TL) models were trained and tested on the captured images using Adam, Sgdm, and RMSprop optimizers. The outcomes of these pre-trained DL models were compared with the custom CNN-LSTM architecture. A learning rate of 0.0002 was found suitable after fine-tuning the model for various learning rates. Finally, the performance evaluation of the various models used is done using the confusion matrix and AUC-ROC curve.

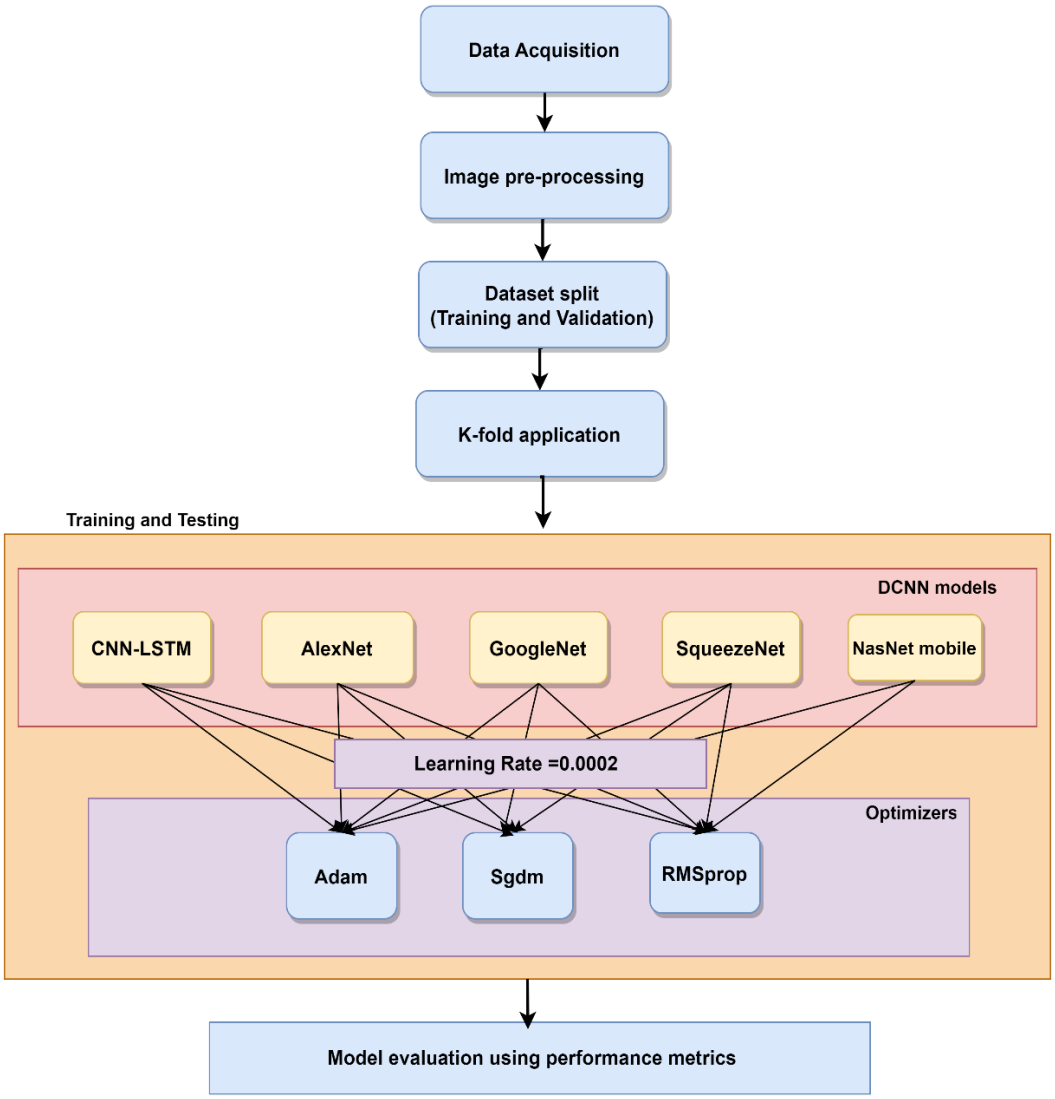


Fig 4.8 Overall system architecture

4.10 Hyper Parameters

The training process of any network is controlled by hyperparameters. They are of two types: model and optimization. Model hyperparameters establish a network's structure, whereas optimization hyper parameters determine the training process's path to the intended output [19].

4.10.1 Batch size

The quantity of training samples that are given to the network in one iteration is known as the batch size [20]. There are three possible batch sizes:

1. Batch mode: In this mode, the iteration and epoch values are similar because the batch size is equal to the complete dataset. Although it takes less time, it uses too much memory [20].
2. Mini-batch mode: In this, the batch size is greater than one but less than the size of the entire dataset. Usually, a sum that can be divided by the size of the entire dataset. If the selected batch size is not too huge, converges quickly with reduced memory demand [20].
3. Stochastic mode: In this mode, there is just one batch. Consequently, after every sample, the gradient and neural network parameters are changed. It is very time-consuming.

4.10.2 Optimizers

Optimizers are algorithms or techniques that alter the weights and learning rates of your neural network to decrease losses. These algorithms minimize the loss function by iteratively modifying the model's parameters. The performance of a deep learning model can be significantly improved by selecting a suitable optimizer and tweaking its hyper parameters.

1. **Adam:** The Adam is a highly effective optimization algorithm frequently employed in deep learning. Adam maintains two moving averages for each parameter, namely the first moment (mean) and the second moment (uncentered variance) of the gradients. These moving averages adaptively fine-tune the learning rates for every parameter during training, making Adam well-suited for non-stationary and noisy data. The algorithm also incorporates bias correction, which helps in the initialization phase of training. Adam provides a robust and efficient optimization method that often converges faster and more reliably. The method is computationally efficient and has little memory requirements [21].
2. **Sgdm:** The Sgdm (Stochastic gradient descent with momentum) improves on the standard stochastic gradient descent by including a momentum factor [22]. This momentum factor is intended to speed up the optimization process by accumulating a moving average of previous gradients and imparting inertia to parameter updates. This smoothens out the oscillations in the optimization path, resulting in faster convergence and higher stability, especially when noisy gradients or ill-conditioned optimization landscapes are present [19].
3. **RMSprop:** The RMSprop (Root Mean Square Propagation) addresses some of the limitations of the basic stochastic gradient descent (SGD) by adapting the learning rates

for individual model parameters [23]. It retains a moving average of the squared gradient and alters the weight updates by this magnitude [24,25]. This adaptive learning rate technique helps mitigate the challenges posed by varying gradients in different dimensions, allowing the algorithm to converge more effectively on complex loss landscapes.

4.10.3 Epoch

Epoch is the number of times that the learning algorithm will work through the entire training dataset. One epoch signifies that each sample in the training dataset has had a chance to alter the internal model parameters. Each epoch is made up of one or more batches. Too few epochs can lead to underfitting, which occurs when the model is unable to capture the patterns in the data. Too many epochs, on the other hand, might lead to overfitting, in which the model becomes too particular to the training data and is unable to generalize to new data.

Table 4. 1 Hyperparameter of the custom CNN-LSTM used for classification

Parameter	Value/Type
Optimizer	Adam/Sgdm/RMSprop
Initial learning rate	0.0002
Batch size	16
Epochs	50

Table 4.1 shows the hyperparameters utilized for the customized CNN-LSTM model used for the classification of dehusked arecanuts into five categories.

4.10.4 Performance Evaluation Metrics

Once the ML model has been developed and the ground truth has been established, it is critical to test its efficacy in describing, forecasting, or evaluating outcomes [26,27].

Accuracy: It is calculated by dividing the number of correctly predicted classes by the total number of samples analyzed.

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TP+FP+FN}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 1$$

Recall: It is used to calculate the percentage of correctly classified positive patterns.

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \quad \text{.....2}$$

Precision: It is used to determine the positive patterns that are successfully predicted by all predicted patterns in a positive class.

$$\text{Precesion} = \frac{\text{TP}}{\text{TP+FP}} \quad \text{.....3}$$

F1 Score: computes the harmonic average of recall and precision rates.

$$\text{F1 Score} = 2\text{x}\frac{\text{Precesion x Recall}}{\text{Precision+Recall}} \quad \text{.....4}$$

Where TN and TP are the number of successfully classified negative and positive cases, respectively, and FN and FP represent the number of misclassified positive and negative cases, respectively.

A high accuracy rate indicates that the model's capacity to discriminate negative samples is adequately represented. The higher the accuracy rate, the better the model's ability to differentiate between positive and negative samples [28].

A high recall rate indicates that the classifier will make every effort to predict positive samples as positive samples. The recall rate can accurately reflect the model's capacity to discriminate between positive and negative samples. The higher the recall rate, the better the model's capacity to distinguish between positive and negative data [28].

To assess the effectiveness or quality of the models utilized, metrics such as the Confusion matrix and the AUC-ROC (Area Under the Curve-Receiver Operating Characteristic) curve are used [29]. ROC is used not only to compare the learning algorithms but also to construct an optimal learning model [30,31]. The AUC-ROC curve is only applicable to binary classification tasks. However, we may extend it to multiclass classification issues by employing the One vs. All technique [32].

4.11 References

1. R. Archana and P. S. E. Jeevaraj, “Deep learning models for digital image processing: a review,” *Artif Intell Rev*, vol. 57, no. 1, p. 11, Jan. 2024
2. W. Jia, Y. Tian, R. Luo, Z. Zhang, J. Lian, and Y. Zheng, “Detection and segmentation of overlapped fruits based on optimized mask R-CNN application in apple harvesting robot,” *Computers and Electronics in Agriculture*, vol. 172, p. 105380, May 2020
3. X. Mai, H. Zhang, X. Jia, and M. Q.-H. Meng, “Faster R-CNN With Classifier Fusion for Automatic Detection of Small Fruits,” *IEEE Trans. Automat. Sci. Eng.*, pp. 1–15, 2020, doi: 10.1109/TASE.2020.2964289.
4. M. Khoshdeli, R. Cong, and B. Parvin, “Detection of nuclei in H&E stained sections using convolutional neural networks,” in *2017 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)*, Orland, FL, USA: IEEE, 2017, pp. 105–108. doi: 10.1109/BHI.2017.7897216.
5. L. Alzubaidi *et al.*, “Review of deep learning: concepts, CNN architectures, challenges, applications, future directions,” *J Big Data*, vol. 8, no. 1, p. 53, Mar. 2021
6. R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, “Convolutional neural networks: an overview and application in radiology,” *Insights Imaging*, vol. 9, no. 4, pp. 611–629, Aug. 2018, doi: 10.1007/s13244-018-0639-9.
7. A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.
8. Szegedy *et al.*, “Going deeper with convolutions,” *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9, Jun. 2015, doi: 10.1109/CVPR.2015.7298594.
9. K. Dang, T. Vo, L. Ngo, and H. Ha, “A deep learning framework integrating MRI image preprocessing methods for brain tumor segmentation and classification,” *IBRO Neuroscience Reports*, vol. 13, pp. 523–532, Dec. 2022, doi: 10.1016/j.ibneur.2022.10.014.
10. E. Yilmaz and M. Trocan, “A modified version of GoogLeNet for melanoma diagnosis,” *Journal of Information and Telecommunication*, vol. 5, no. 3, pp. 395–405, Jul. 2021, doi: 10.1080/24751839.2021.1893495.
11. “Image Classification With MobileNet,” Built In. Accessed: Oct. 22, 2024. [Online]. Available: https://builtin.com/machine-learning/mobilenet

12. Khasoggi, E. Ermatita, and S. Samsuryadi, “Efficient mobilenet architecture as image recognition on mobile and embedded devices,” *IJEECS*, vol. 16, no. 1, p. 389, Oct. 2019, doi: 10.11591/ijeecs.v16.i1.pp389-394.
13. B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, “Learning Transferable Architectures for Scalable Image Recognition,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT: IEEE, Jun. 2018, pp. 8697–8710. doi: 10.1109/CVPR.2018.00907.
14. H. C. Reis and V. Turk, “COVID-DSNet: A novel deep convolutional neural network for detection of coronavirus (SARS-CoV-2) cases from CT and Chest X-Ray images,” *Artificial Intelligence in Medicine*, vol. 134, p. 102427, Dec. 2022, doi: 10.1016/j.artmed.2022.102427.
15. F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size,” 2016, *arXiv*. doi: 10.48550/ARXIV.1602.07360.
16. M. Tsivgoulis, T. Papastergiou, and V. Megalooikonomou, “An improved SqueezeNet model for the diagnosis of lung cancer in CT scans,” *Machine Learning with Applications*, vol. 10, p. 100399, Dec. 2022, doi: 10.1016/j.mlwa.2022.100399.
17. J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL: IEEE, Jun. 2009, pp. 248–255. doi: 10.1109/CVPR.2009.5206848.
18. C. Kim, F. Li, and J. M. Rehg, “Multi-object Tracking with Neural Gating Using Bilinear LSTM,” in *Computer Vision – ECCV 2018*, vol. 11212, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds., Cham: Springer International Publishing, 2018, pp. 208–224. doi: 10.1007/978-3-030-01237-3_13.
19. P. Sharma and R. S. Anand, “A comprehensive evaluation of deep models and optimizers for Indian sign language recognition,” *Graphics and Visual Computing*, vol. 5, p. 200032, Dec. 2021, doi: 10.1016/j.gvc.2021.200032.
20. A. Murphy and F. Gaillard, “Batch size (machine learning),” in *Radiopaedia.org*, Radiopaedia.org, 2017. doi: 10.53347/rID-56140.
21. D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” 2014, *arXiv*. doi: 10.48550/ARXIV.1412.6980.

22. B. T. Polyak, “Some methods of speeding up the convergence of iteration methods,” *USSR Computational Mathematics and Mathematical Physics*, vol. 4, no. 5, pp. 1–17, Jan. 1964, doi: 10.1016/0041-5553(64)90137-5.
23. Hinton G, Srivastava N, Swersky K (2012) Neural Networks for Machine Learning
24. E. Hassan, M. Y. Shams, N. A. Hikal, and S. Elmougy, “The effect of choosing optimizer algorithms to improve computer vision tasks: a comparative study,” *Multimed Tools Appl*, vol. 82, no. 11, pp. 16591–16633, May 2023, doi: 10.1007/s11042-022-13820-0.
25. R. Zaheer and H. Shaziya, “A Study of the Optimization Algorithms in Deep Learning,” in *2019 Third International Conference on Inventive Systems and Control (ICISC)*, Coimbatore, India: IEEE, Jan. 2019, pp. 536–539. doi: 10.1109/ICISC44355.2019.9036442.
26. R. Boutaba *et al.*, “A comprehensive survey on machine learning for networking: evolution, applications and research opportunities,” *J Internet Serv Appl*, vol. 9, no. 1, p. 16, Dec. 2018, doi: 10.1186/s13174-018-0087-2.
27. M. Carbonero-Ruz, F. J. Martínez-Estudillo, F. Fernández-Navarro, D. Becerra-Alonso, and A. C. Martínez-Estudillo, “A two-dimensional accuracy-based measure for classification performance,” *Information Sciences*, vol. 382–383, pp. 60–80, Mar. 2017, doi: 10.1016/j.ins.2016.12.005.
28. B. Li, “Hearing loss classification via AlexNet and extreme learning machine,” *International Journal of Cognitive Computing in Engineering*, vol. 2, pp. 144–153, Jun. 2021, doi: 10.1016/j.ijcce.2021.09.002.
29. V. Mingote, A. Miguel, A. Ortega, and E. Lleida, “Optimization of the area under the ROC curve using neural network supervectors for text-dependent speaker verification,” *Computer Speech & Language*, vol. 63, p. 101078, Sep. 2020, doi: 10.1016/j.csl.2020.101078.
30. T. Fawcett, “An introduction to ROC analysis,” *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, Jun. 2006, doi: 10.1016/j.patrec.2005.10.010.
31. Jin Huang and C. X. Ling, “Using AUC and accuracy in evaluating learning algorithms,” *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 3, pp. 299–310, Mar. 2005, doi: 10.1109/TKDE.2005.50.
32. D. J. Hand and R. J. Till, “[No title found],” *Machine Learning*, vol. 45, no. 2, pp. 171–186, 2001, doi: 10.1023/A:1010920819831.

CHAPTER 5

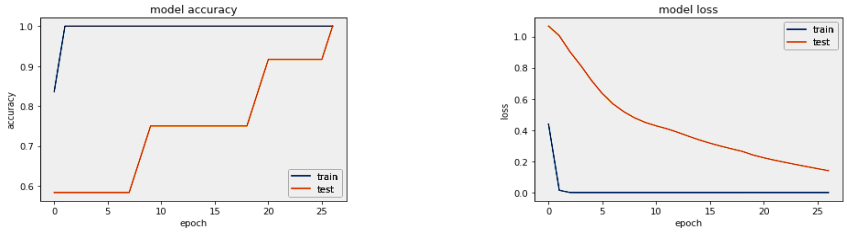
RESULTS AND ANALYSIS

This chapter presents the results of arecanut classification using DL models for three different cases. In Case 1, the Custom CNN model is compared with pre-trained MobileNet. In Case 2, the Custom CNN is compared with the pre-trained AlexNet model, and in Case 3, the Custom CNN-LSTM model is compared with four pre-trained algorithms viz AlexNet, GoogLeNet, SqueezeNet, and NasNetMobile.

All these models were implemented successfully using MATLAB on an Intel i3 Core computer with an NVIDIA PCIc card having TU116 GPU with 6GB GDDR6 dedicated graphics RAM and 16 GB main RAM.

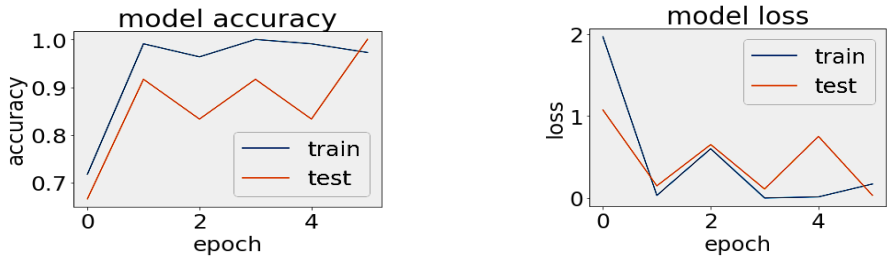
5.1 Binary classification: Custom CNN Vs MobileNet (Case 1)

In this section, the performance of the custom CNN and MobileNet models was evaluated on augmented and non-augmented images and then the outcome was compared. For this purpose, a dataset of 120 arecanut images (60 each) of healthy and diseased nuts was utilized. This dataset was created using the image acquisition system described in section 3.1 in chapter 3. The confusion matrix was used to evaluate the performance of the models, and the 10-fold cross-validation method was applied to cross-validate the models.



a) CNN Training and validation accuracy **b) CNN Training and validation loss**

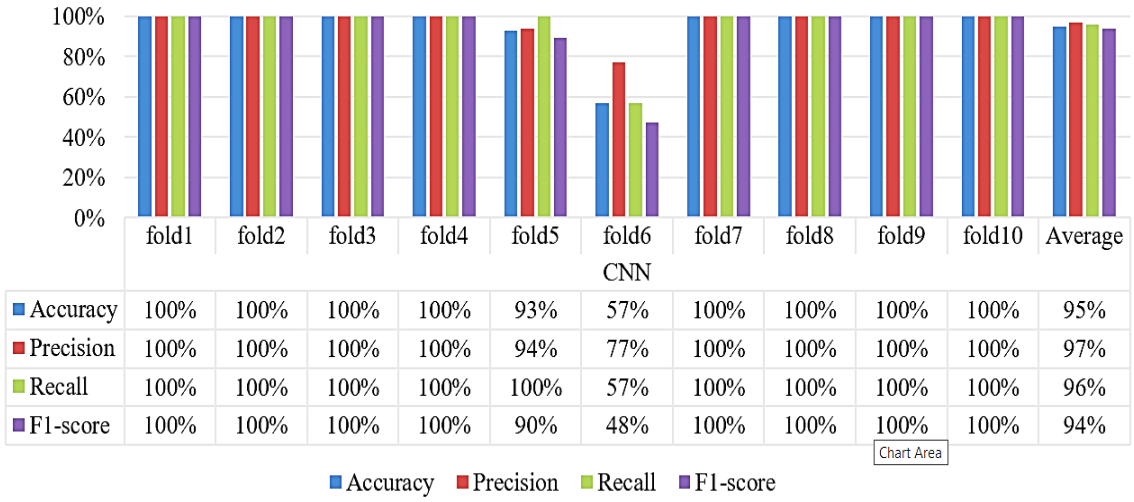
Fig. 5.1: CNN training and validation accuracy and training and validation loss



a) Training and validation accuracy **b) Training and validation loss**

Fig. 5.2: MobileNet training and validation accuracy and training and validation loss

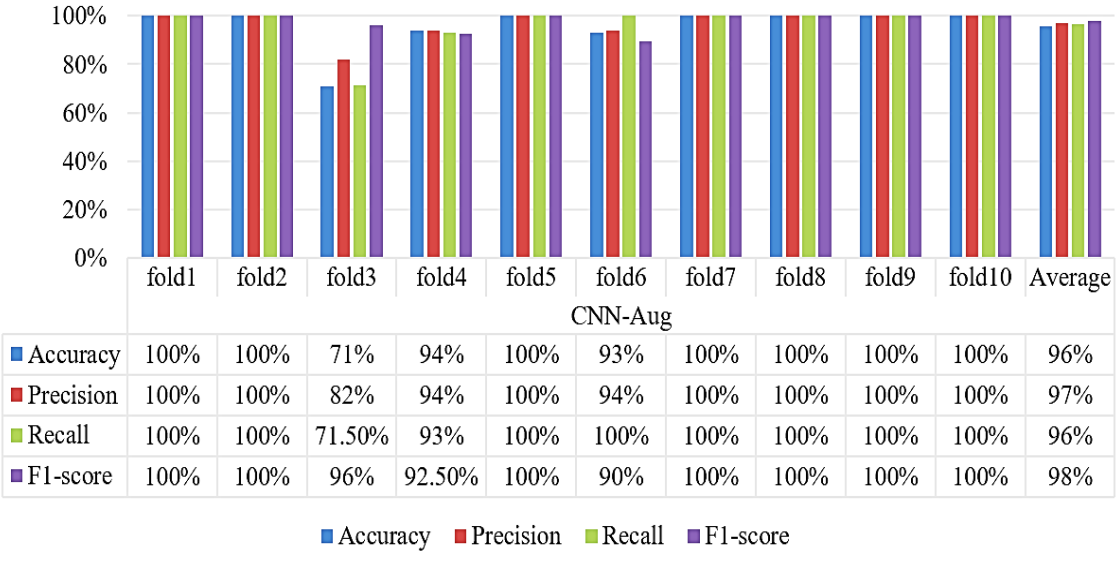
Fig. 5.1 and Fig 5.2 indicate the CNN and MobileNet training and validation accuracy and training and validation loss, respectively. We train a CNN model with 10-fold cross-validation with and without data augmentation for each database.



CNN	fold1	fold2	fold3	fold4	fold5	fold6	fold7	fold8	fold9	fold10	Average
Accuracy	100%	100%	100%	100%	93%	57%	100%	100%	100%	100%	95%
Precision	100%	100%	100%	100%	94%	77%	100%	100%	100%	100%	97%
Recall	100%	100%	100%	100%	100%	57%	100%	100%	100%	100%	96%
F1-score	100%	100%	100%	100%	90%	48%	100%	100%	100%	100%	94%

Fig. 5.3: Various metrics for the CNN model without data augmentation

Figure 5.3 shows the use of the 10-fold method for CNN without augmentation. An average accuracy, precision, recall, and F1-score of 95%, 97%, 96%, and 94%, respectively, were achieved.



CNN-Aug	fold1	fold2	fold3	fold4	fold5	fold6	fold7	fold8	fold9	fold10	Average
Accuracy	100%	100%	71%	94%	100%	93%	100%	100%	100%	100%	96%
Precision	100%	100%	82%	94%	100%	94%	100%	100%	100%	100%	97%
Recall	100%	100%	71.50%	93%	100%	100%	100%	100%	100%	100%	96%
F1-score	100%	100%	96%	92.50%	100%	90%	100%	100%	100%	100%	98%

Fig. 5.4: Various metrics for the CNN model with data augmentation

Figure 5.4 shows the use of the 10-fold method for CNN with augmentation technique. An average accuracy, precision, recall, and F1-score of 96%, 97%, 96%, and 98%, respectively, were achieved.

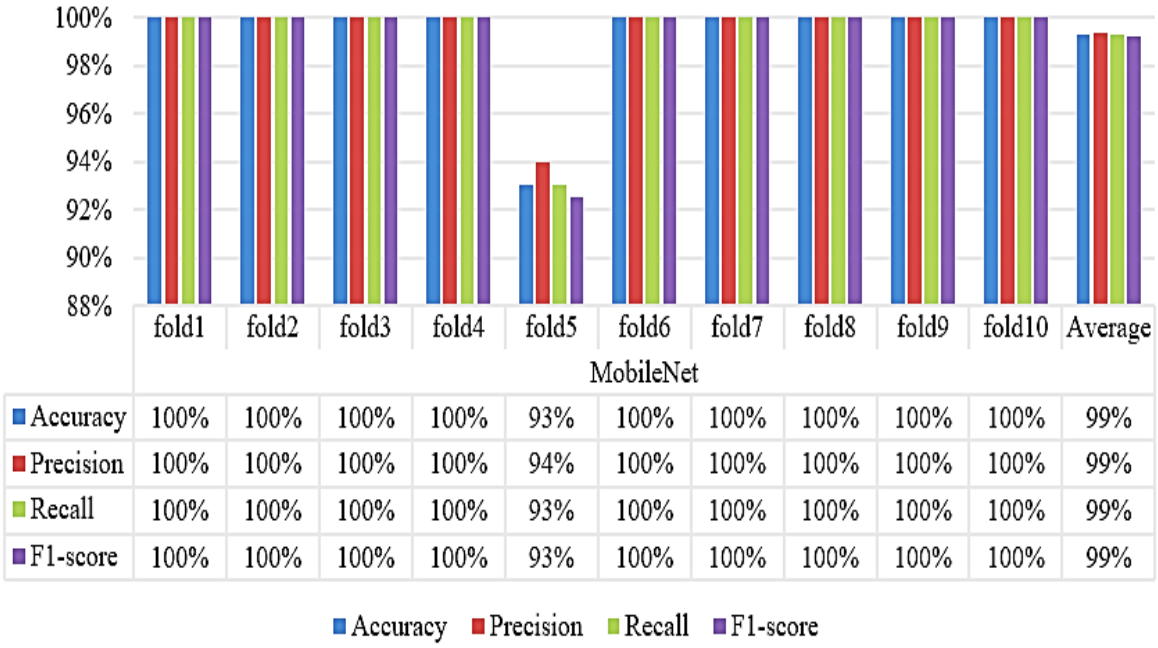


Fig. 5.5: Various metrics for the MobileNet model without data augmentation

Figure 5.5 shows the use of the 10-fold method for the MobileNet network without the augmentation technique. An average accuracy, precision, recall, and F1-score of 99%, respectively, were achieved.

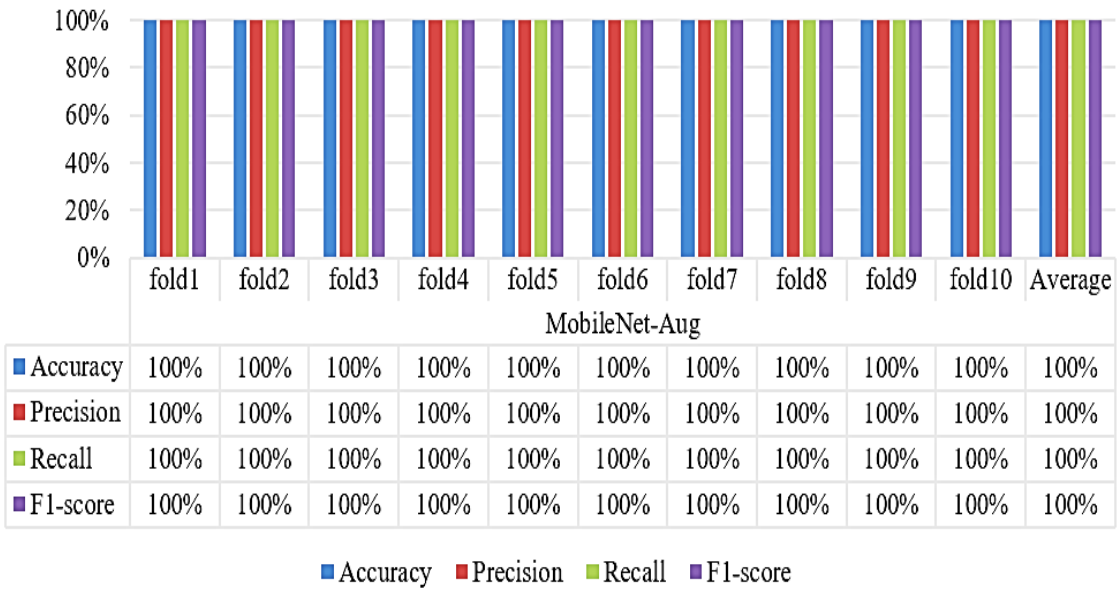


Fig. 5.6: Various metrics for the MobileNet model with data augmentation

Figure 5.6 shows the use of the 10-fold for the MobileNet network with augmentation technique. An average accuracy, precision, recall, and F1-score of 100%, respectively, were achieved.

5.2 Multi-class classification: Custom CNN Vs AlexNet (Case 2)

This section includes testing the classification of de-husked arecanuts into five categories, namely Supari, Kharad, Tukada, Laal, and Baad, using customized custom CNN and standard AlexNet Models. A 5 and 10-fold cross-validation method was utilized for both the CNN and AlexNet models. For this purpose, a dataset of 300 arecanut images (60 each) of above mentioned five categories was created using an image acquisition system described in section 3.1 from Chapter 3.

5.2.1 Training and Validation Accuracy and Loss

Figures 5.7, 5.8, 5.9, and 5.10 show the training and validation accuracy with loss curves against the number epochs for CNN(k=5,10) and AlexNet (k=5,10) models, respectively. The hyperparameters tuned are: Learning Rate=0.0002, Optimizer=Adam, Batch size: 16, epoch=50.

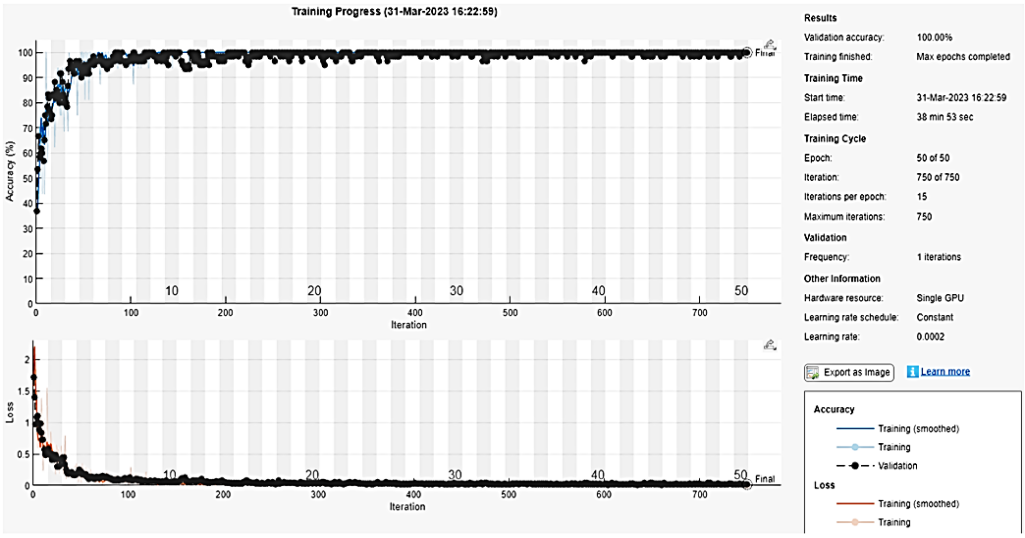


Fig 5.7: Training and validation accuracy and loss curves for CNN (5-fold)

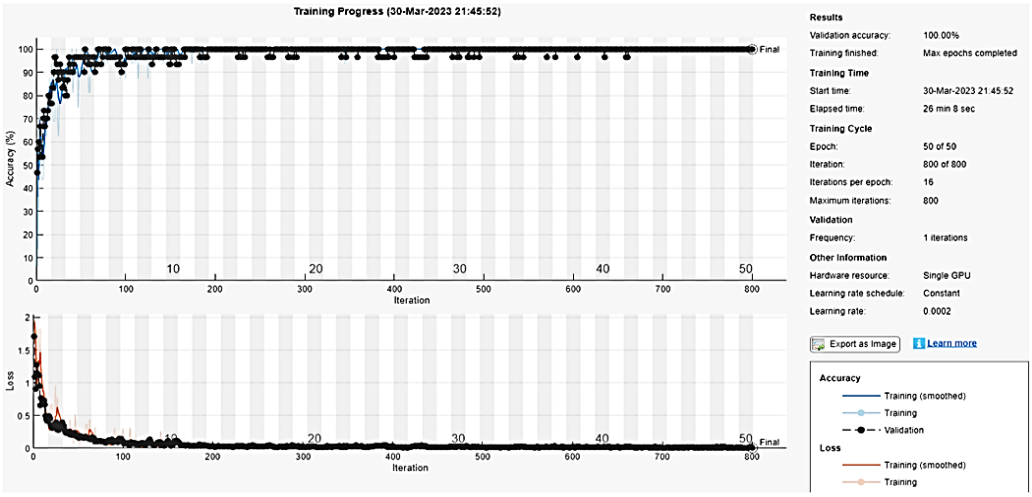


Fig 5.8: Training and validation accuracy and loss curves for CNN (10-fold)

Figures 5.7 and 5.8 show that the training-validation accuracy stabilizes and follows each other after 15 epochs for CNN.

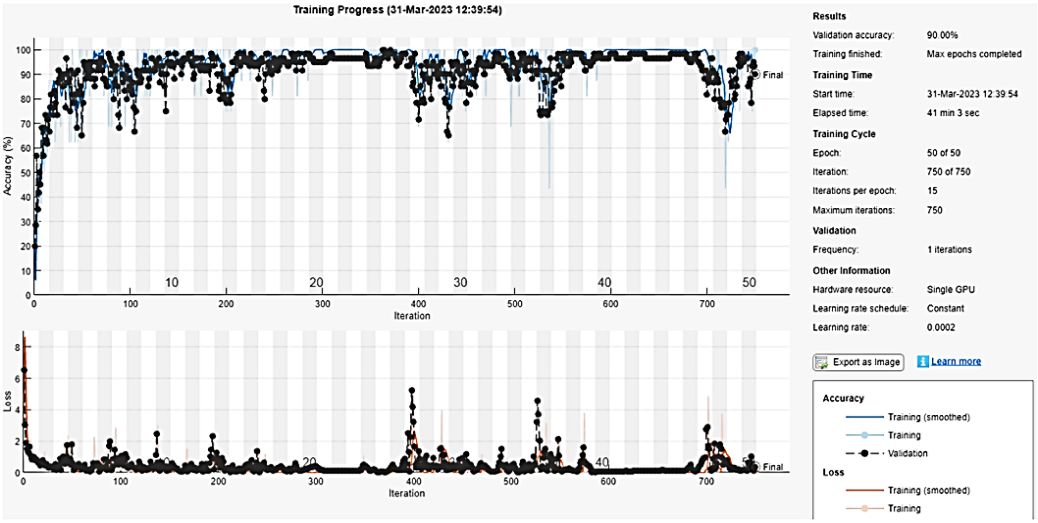


Fig 5.9: Training and validation accuracy and loss curves for AlexNet (5-fold)

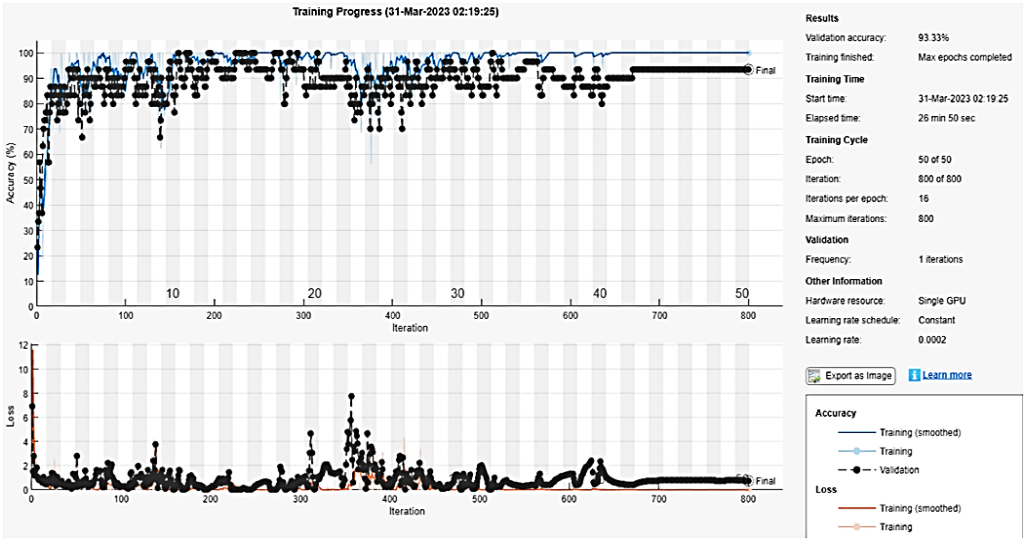
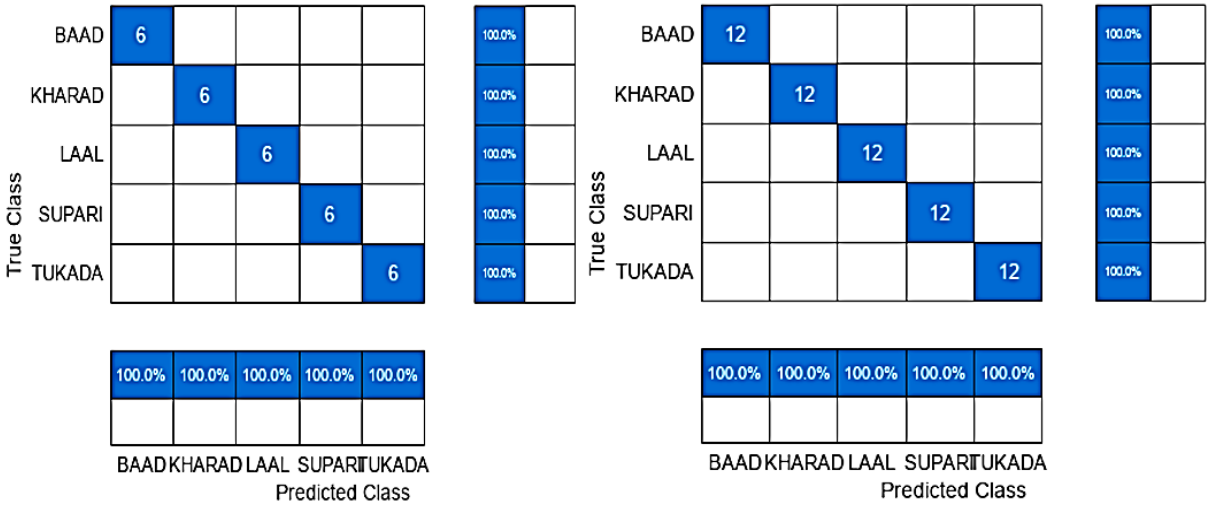


Fig 5.10: Training and validation accuracy and loss curves for AlexNet (10-fold)

For the AlexNet model, the distribution fluctuates during entire folds, as shown in Figures 5.9 and 5.10. The training and validation loss for CNN is low compared to AlexNet. It is observed that the training-validation loss distribution is uneven for AlexNet. So, the designed CNN model performs better than the standard AlexNet model.

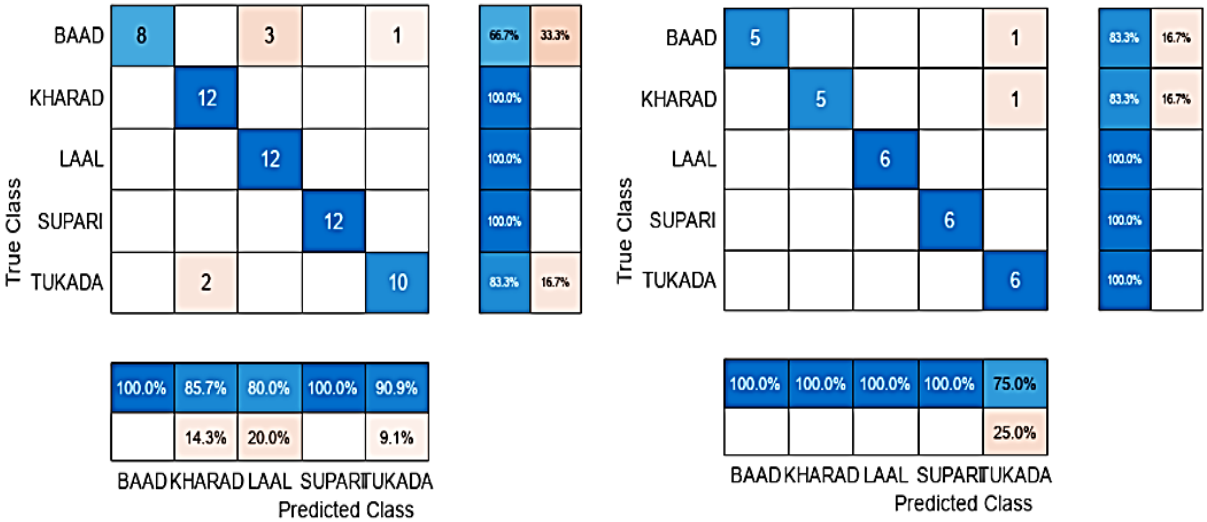
5.2.2 Confusion Matrix:

The comparative performance analysis of both models was checked using a confusion matrix. The confusion matrix for one of the 5/10-fold cross-validation for CNN and AlexNet is shown in Figure 5.11 and Figure 5.12, respectively.



a) 5-fold **b) 10-fold**

Fig 5. 11: Confusion matrix chart for CNN for 5-fold and 10-fold



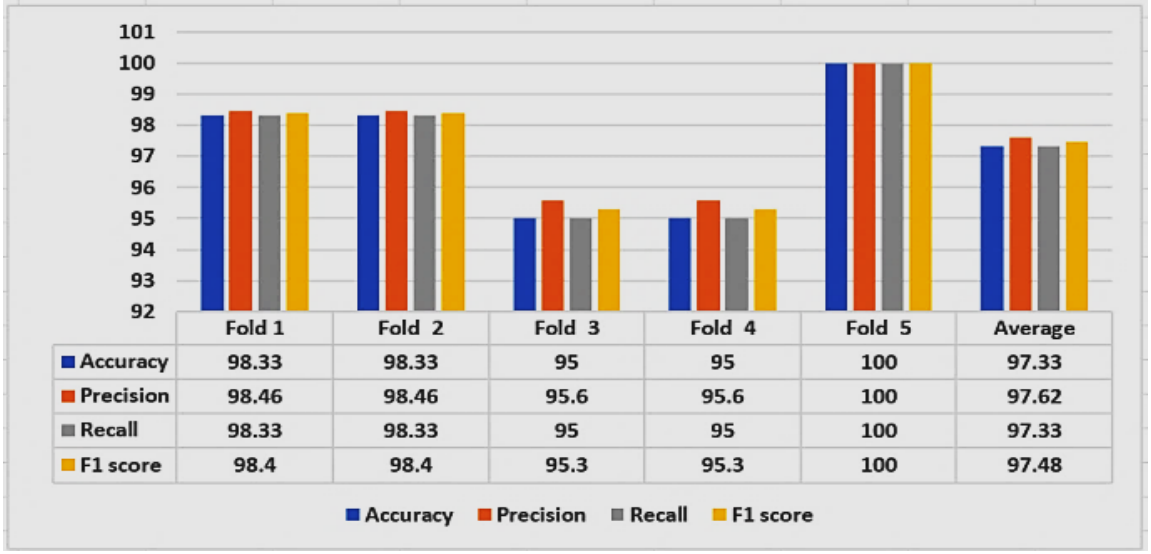
a) 5-fold **b) 10-fold**

Fig 5.12: Confusion matrix chart for AlexNet for 5-fold and 10-fold

From Figure 5.11 (a) and (b), the classification accuracy is 100 %, indicating that all five classes are correctly classified. For 5-fold, 60 images are used for validation (12 from each class) and 240 for training the model. Whereas for 10-fold, 30 are used for validation (6 from each class), and 270 are used for training. As shown in Figure 5.12 (a), it is seen that Baad and Tukada are wrongly classified, thereby decreasing the accuracy to 90% for that fold. Similarly,

from Figure 5.12 (b), for 10-fold, Baad and Kharad category nuts are classified wrongly, which gives an accuracy of 93.33%.

Fig 5.13, 5.14, 5.15, and 5.16 show various performance matrices for CNN and AlexNet models for 5 folds and 10 folds, respectively.



	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Average
Accuracy	98.33	98.33	95	95	100	97.33
Precision	98.46	98.46	95.6	95.6	100	97.62
Recall	98.33	98.33	95	95	100	97.33
F1 score	98.4	98.4	95.3	95.3	100	97.48

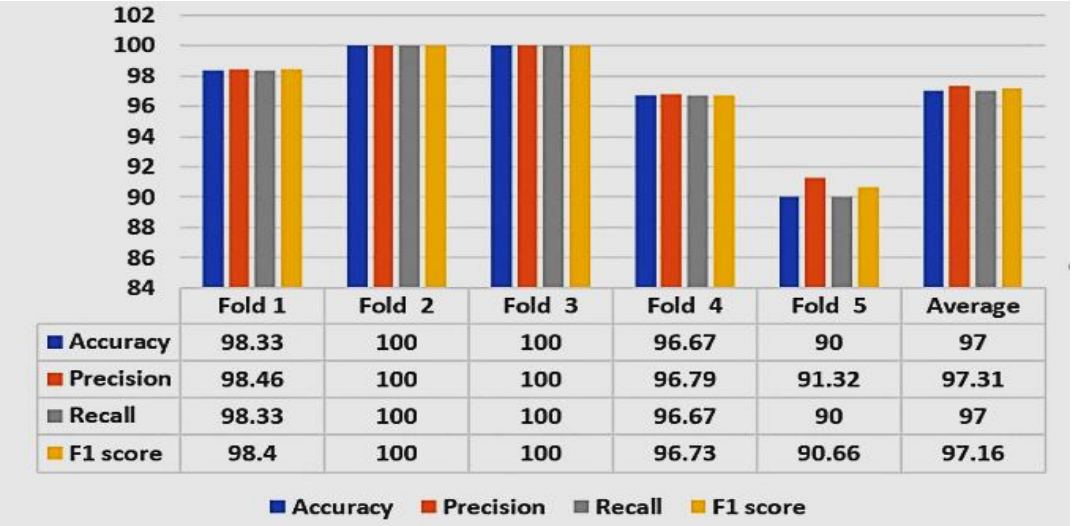
Fig 5.13: Various metrics for the CNN model for 5-fold

From Figure 5.13, it is seen that the accuracy, precision, recall, and F1 scores are 97.33%, 97.62%, 97.33%, and 97.48 %, respectively, for the 5-fold customized CNN model.

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	Average
Accuracy	96.67	96.67	96.67	96.67	100	100	100	100	100	96.67	98.34
Precision	97.14	97.14	97.14	97.14	100	100	100	100	100	97.14	98.57
Recall	96.67	96.67	96.67	96.67	100	100	100	100	100	96.67	98.34
F1 score	96.9	96.9	96.9	96.9	100	100	100	100	100	96.9	98.45

Fig 5.14: Various metrics for the CNN model for 10-fold

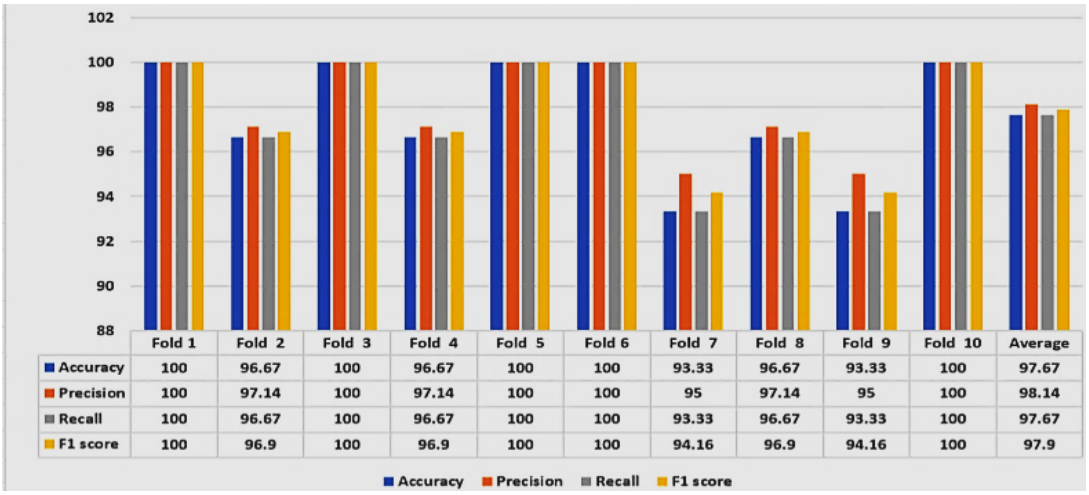
From Figure 5.14, it is seen that the accuracy, precision, recall, and F1 scores are 98.34%, 98.57%, 98.34%, and 98.45 %, respectively, for the 10-fold customised CNN model.



	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Average
Accuracy	98.33	100	100	96.67	90	97
Precision	98.46	100	100	96.79	91.32	97.31
Recall	98.33	100	100	96.67	90	97
F1 score	98.4	100	100	96.73	90.66	97.16

Fig 5.15: Various metrics for the AlexNet model for 5-fold

From Figure 5.15, it is seen that the accuracy, precision, recall, and F1 scores are 97%, 97.31%, 97%, and 97.16 %, respectively, for the 5-fold AlexNet model.



	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Fold 6	Fold 7	Fold 8	Fold 9	Fold 10	Average
Accuracy	100	96.67	100	96.67	100	100	93.33	96.67	93.33	100	97.67
Precision	100	97.14	100	97.14	100	100	95	97.14	95	100	98.14
Recall	100	96.67	100	96.67	100	100	93.33	96.67	93.33	100	97.67
F1 score	100	96.9	100	96.9	100	100	94.16	96.9	94.16	100	97.9

Fig 5.16: Various metrics for the AlexNet model for 10-fold

From Figure 5.16, it is seen that the accuracy, precision, recall, and F1 scores are 97.67%, 98.14%, 97.67%, and 97.90 %, respectively, for the 10-fold AlexNet model.

The summary of results for CNN and AlexNet models is depicted in Table 5.1.

Table 5.1: Summary of All Matrices for CNN and AlexNet

Model / Metrics		Average			
		Accuracy	Precision	Recall	F1 score
CNN	5-fold	97.33	97.62	97.33	97.48
	10-fold	98.34	98.57	98.34	98.45
AlexNet	5-fold	97.00	97.31	97.00	97.16
	10-fold	97.67	98.14	97.67	97.90

From Table 5.1, it is seen that CNN attains the highest accuracy / F1 score of 97.33% / 97.48% for 5-fold and 98.34% /98.45% for 10-fold, respectively. As the fold is reduced from 10 to 5, it is observed that the CNN model performs equally well.

5.2.3 AUC-ROC

The ROC curve for both the models for 5 and 10 fold is shown in Figure 12. The AUC values for the CNN model are highest for 5-fold and 10-fold.

Figures 5.17 (a) and (b) show the ROC curves for the CNN model with 100% accurate performance. At the same time, Figures 5.18 (a) and (b) show the 90% (5-fold) and 93.33% (10-fold) accurate performance for AlexNet. The ROC curve is a graphical representation of the ML models' binary-class or multi-class testing skills. The ROC curve should be closer to the upper left for improved classification performance. From Figure 5.17 (a) and (b), all the classes of arecanuts achieved an AUC of 1, which shows excellent performance of the CNN model. On the other hand, from Figure 5.18 (a) and (b), it is seen that the value of AUC for Baad (0.94) and Tukada (0.99) for 5-fold, and Baad (0.98) and Kharad (0.99) for 10-fold in case of AlexNet respectively. This shows that CNN performs very well compared to AlexNet for arecanut classification.

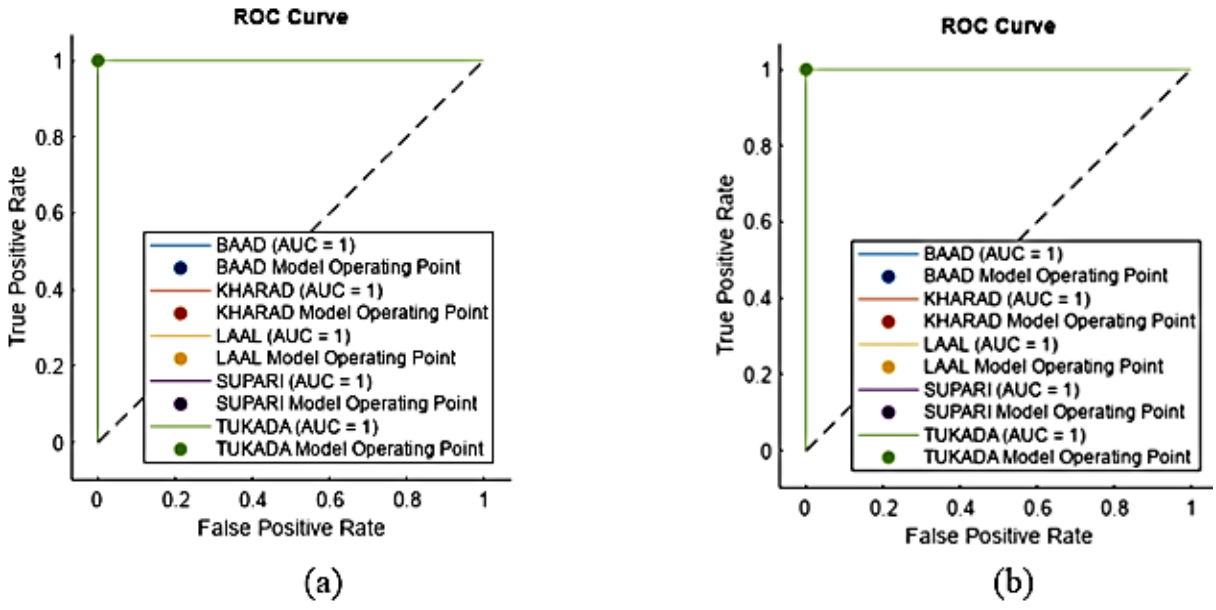
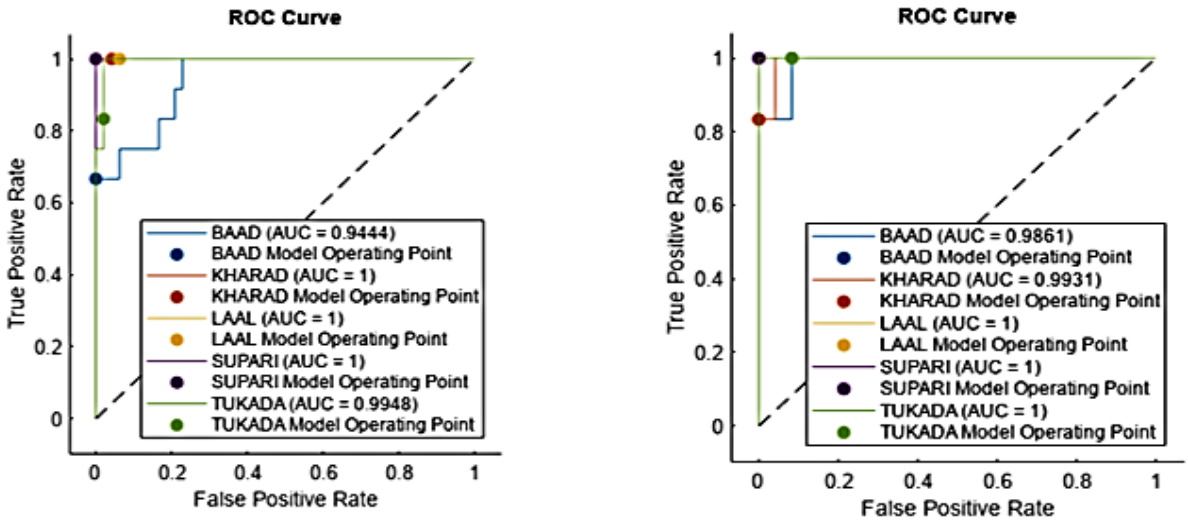


Figure 5.17: AUC- ROC curve of five classes for (a) CNN-5-fold (b) CNN-10-fold



(a) **(b)**

Figure 5.18: AUC- ROC curve of five classes for (a) AlexNet-5-fold (b) AlexNet-10-fold

5.3 Custom CNN-LSTM Vs Pre-Trained Models (Case 3)

In this section, we compare our custom classification model (CNN-LSTM) with the other four pre-trained models described in Chapter 4. In this section, we have used a total of five DCNN transfer-learning models, i.e., AlexNet, GoogLeNet, SqueezeNet, NasNetMobile, and CNN-LSTM, and evaluated their performance with Adam, Sgdm, and RMSprop optimizer. For this purpose, a dataset of 300 arecanut images (60 each) of five categories was created using an image acquisition system described in section 3.1 from Chapter 3. The k-fold cross-validation approach is often used to evaluate the performance of a DL algorithm on a new dataset for a multi-class problem. This method involves randomly splitting a data set into k nearly equal-sized disjoint folds and then using each fold to test the model that a classification algorithm deduced from the previous k folds [1, 2, 3]. In DL, selecting the k value for cross-validation of a model is crucial. The model's accuracy rises with the amount of k as more training samples are given to it, increasing accuracy. Many scholars have investigated the influence of varying k values on model performance estimation [3]. In this thesis, we have used a five-fold cross-validation method for the estimation of model performance. The accuracy of the algorithm is found by taking an average of all the five folds [4]. Among all the 5-DCNN models, CNN-LSTM gives the highest accuracy, and hence, the performance metrics only for this model are provided below.

Fig 5.19, Fig 5.20, and Fig 5.21 show the training and validation accuracy and loss curves for the CNN-LSTM model for Adam, Sgdm, and RMSprop Optimizer respectively.

From Fig 5.19, it is noticed that the training and validation accuracy and loss settle down within 20 epochs to their maximum average validation accuracy of 98.33 %. However, there is noise present during the epochs for almost all the folds. For some of the folds, it is seen that the training and validation accuracy and loss curves do not follow each other. From Fig 5.20 it is noticed that the training and validation accuracy and loss follow each other after 25 epochs. The noise is not present throughout the 50 epochs. Also, for all the folds, it is noticed that the training and validation accuracy and loss curves follow each other.

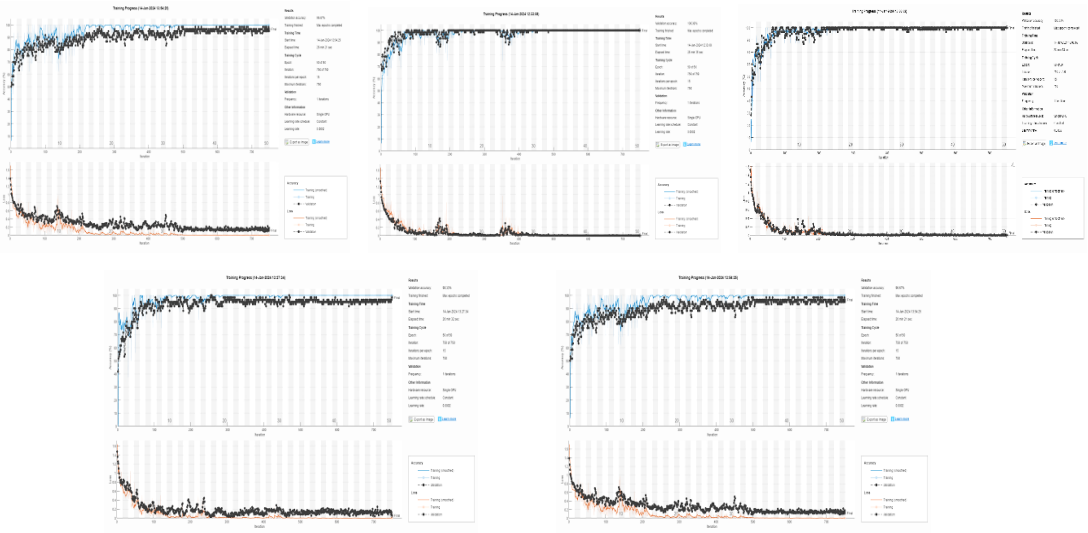


Fig 5.19 The training and validation accuracy and loss curves of CNN-LSTM for Adam Optimizer for 5 folds

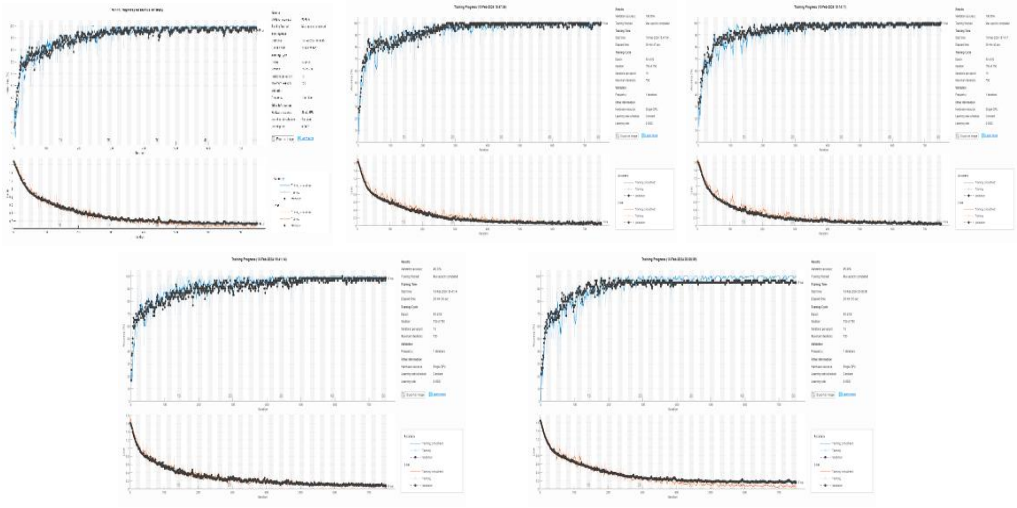


Fig 5.20 The training and validation accuracy and loss curves of CNN-LSTM for Sgdm Optimizer for 5 folds

From Fig 5.21 it is noticed that across all the folds, the noise is present throughout all epochs. The model stabilizes after 15 epochs from its maximum average validation accuracy of

99%. For some of the folds, it is observed that the training and validation accuracy and loss curves follow each other.

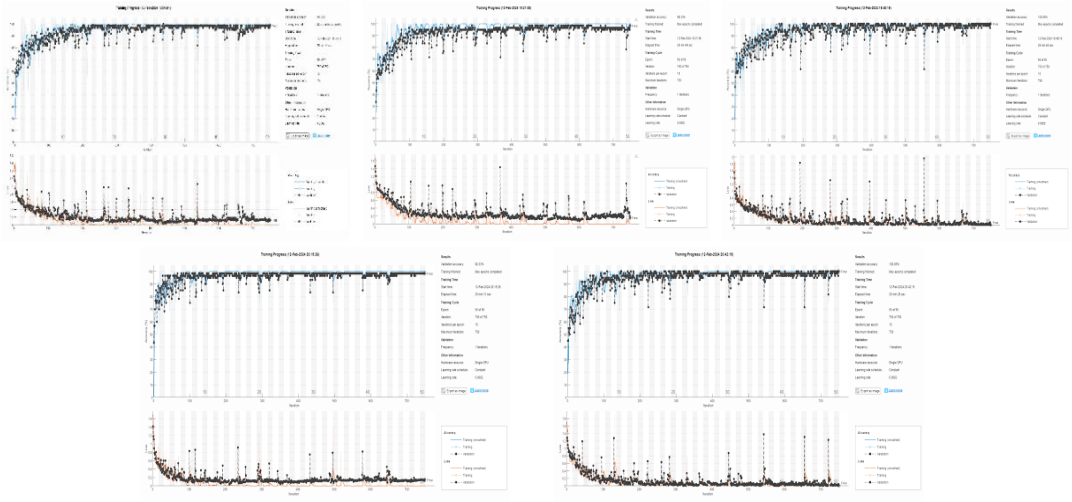


Fig 5.21 The training and validation accuracy and loss curves of CNN-LSTM for RMSprop Optimizer for 5 folds

Fig 5.22, Fig 5.23, and Fig 5.24 show the confusion metrics chart for the CNN-LSTM model for Adam, Sgdm, and RMSprop Optimizer, respectively. It is noted from these figures that out of 5 folds, for two folds the classification is 100 %. For three folds "BAAD" is wrongly classified for Adam and Sgdm optimizer. Whereas across all the optimizers, "KHARAD" is wrongly classified in one of the folds. For Adam and Sgdm, "TUKADA" is wrongly classified in one of the folds. For the RMSprop optimizer, "BAAD" is wrongly classified only twice, which indicates that CNN-LSTM with RMSprop works better for arecanut classification.

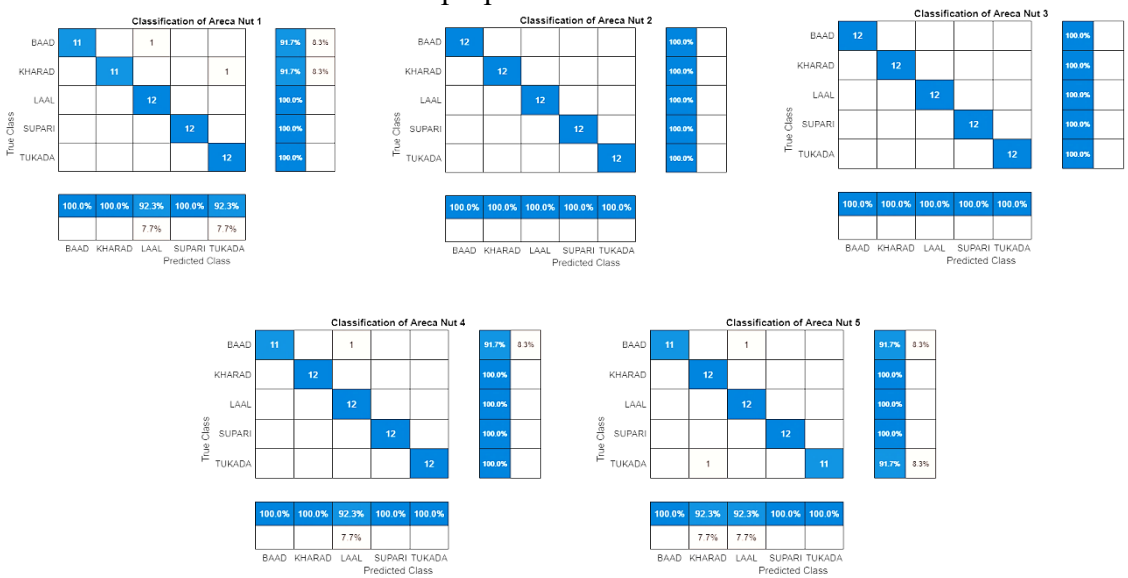


Fig 5.22 Confusion Matrix chart of CNN-LSTM for Adam Optimizer for 5 folds

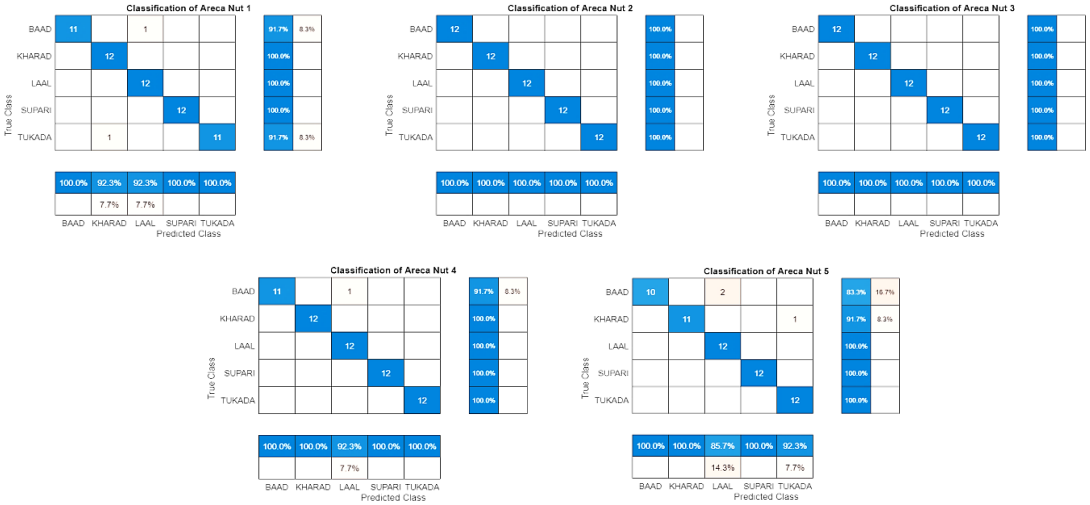


Fig 5.23 Confusion Matrix chart of CNN-LSTM for Sgdm Optimizer for 5 folds

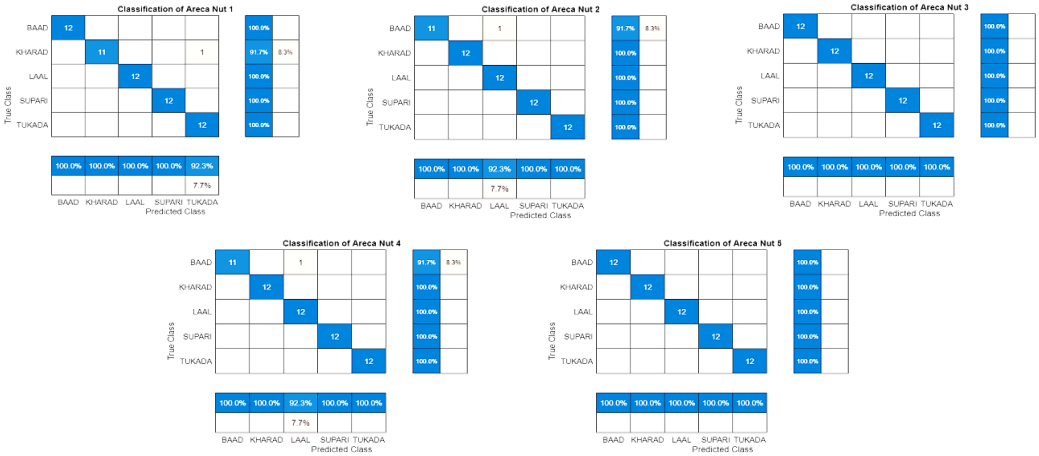


Fig 5.24 Confusion Matrix chart of CNN-LSTM for RMSprop Optimizer for 5 folds

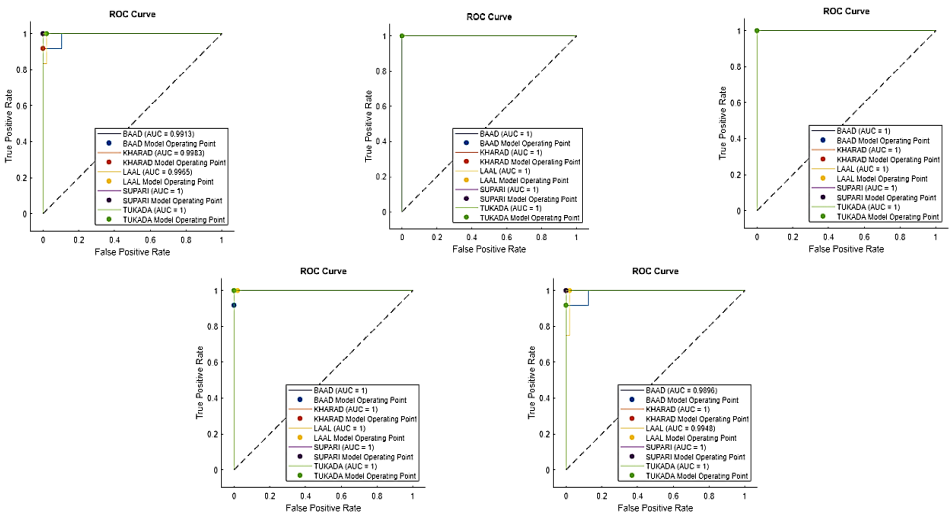


Fig 5.25 AUC- ROC curve of CNN-LSTM for Adam Optimizer for 5 folds

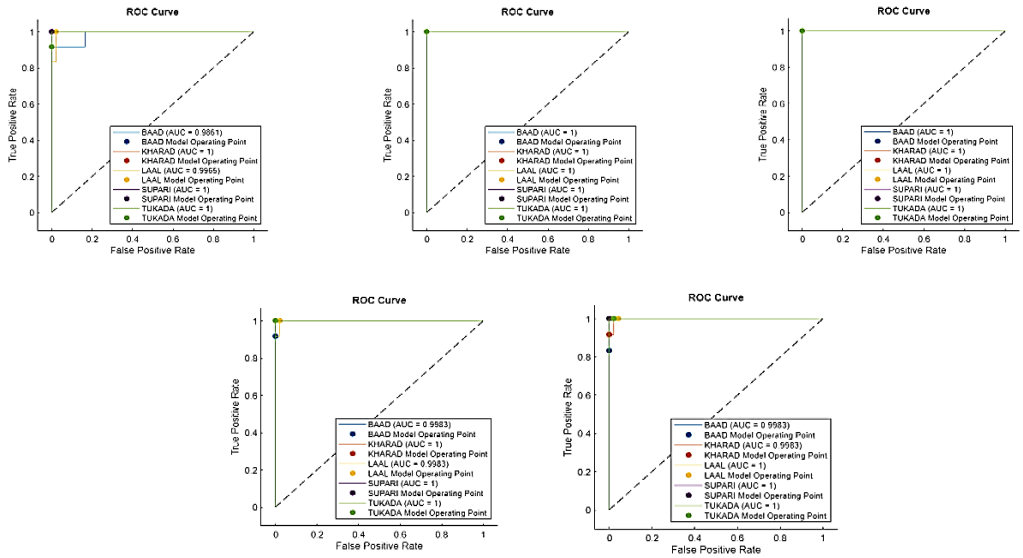


Fig 5.26 AUC- ROC curve of CNN-LSTM model for Sgdm Optimizer for 5 folds

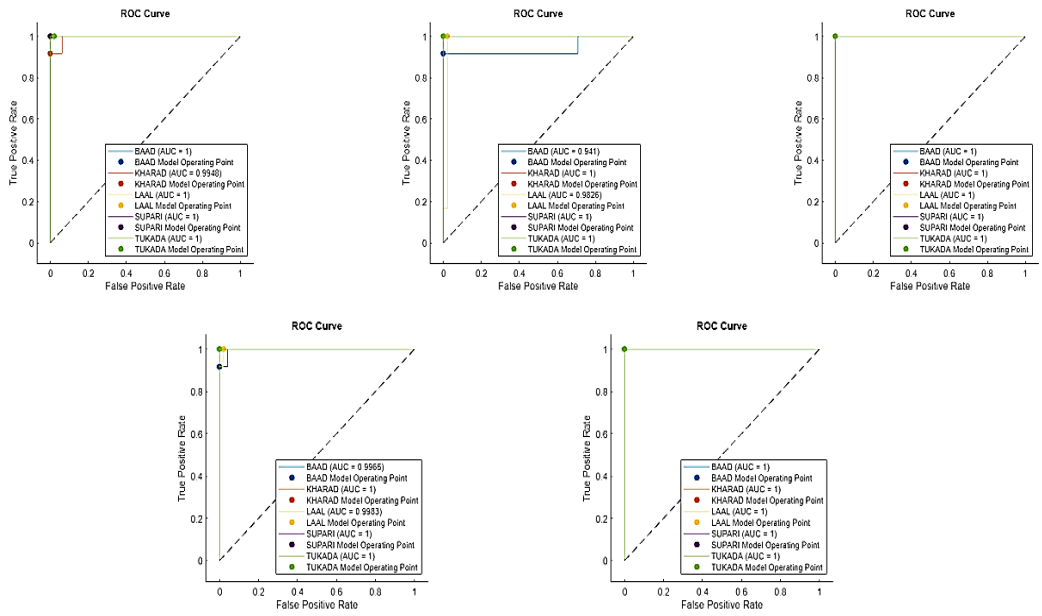


Fig 5.27 AUC- ROC curve of CNN-LSTM model for RMSprop Optimizer for 5 folds

Fig 5.25, Fig 5.26, and Fig 5.27 show the AUC-ROC curves for the CNN-LSTM model for Adam, Sgdm, and RMSprop Optimizer respectively. In all curves, it is noticed that the classification is close to 100 %.

The comparative bar charts of performance measures like accuracy, precision, recall, and F1-score for all five DCNN models with Adam, Sgdm, and RMSprop optimizers for 5-fold cross-validation are presented in Fig. 5.28, Fig. 5.29, and Fig. 5.30 respectively.

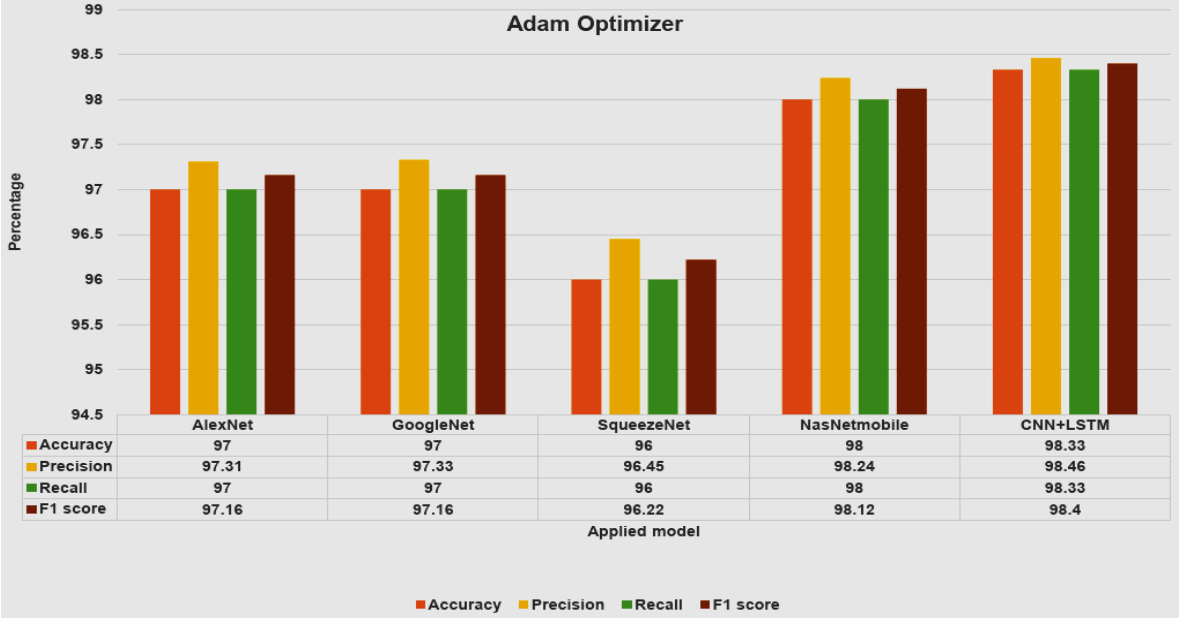


Fig 5.28 Comparison of various metrics for 5-fold cross-validation with different DL models for Adam Optimizer

From Figure 5.28, it is seen that the accuracy, precision, recall, and F1 scores are 98.33%, 98.46%, 98.33%, and 98.4%, respectively, for the 5-fold CNN-LSTM model. Among all the models for Adam optimizer, CNN-LSTM performs better.

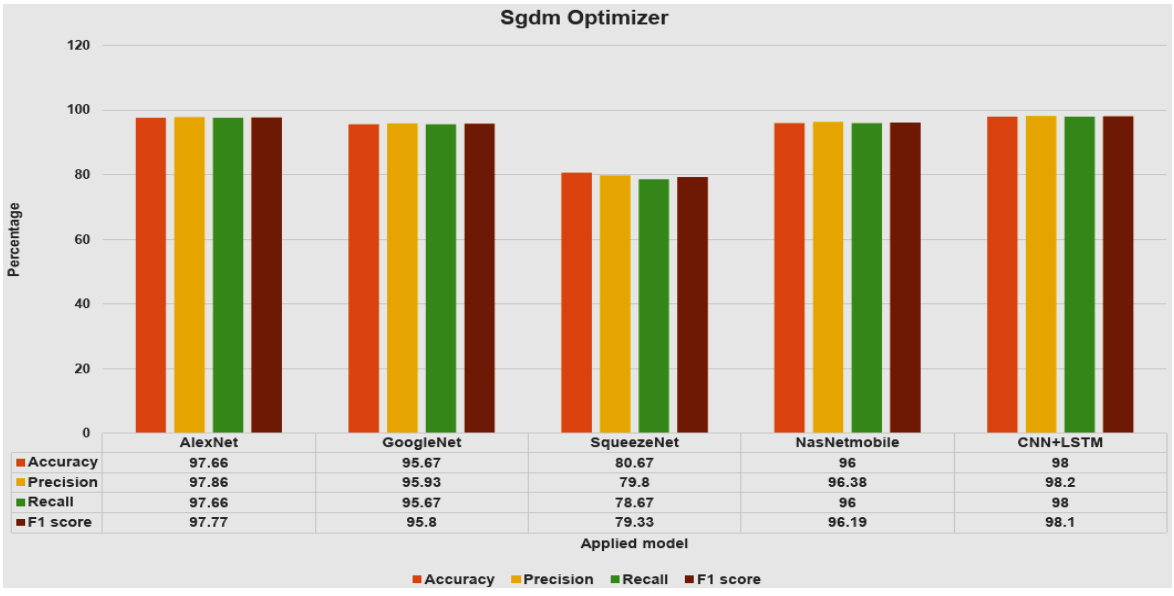


Fig 5.29 Comparison of various metrics for 5-fold cross-validation with different DL models for Sgdm Optimizer

From Figure 5.29, it is seen that the accuracy, precision, recall, and F1 scores are 98%, 98.2%, 98%, and 98.1%, respectively, for the 5-fold CNN-LSTM model. Among all other models for Sgdm optimizer, CNN-LSTM performs better.

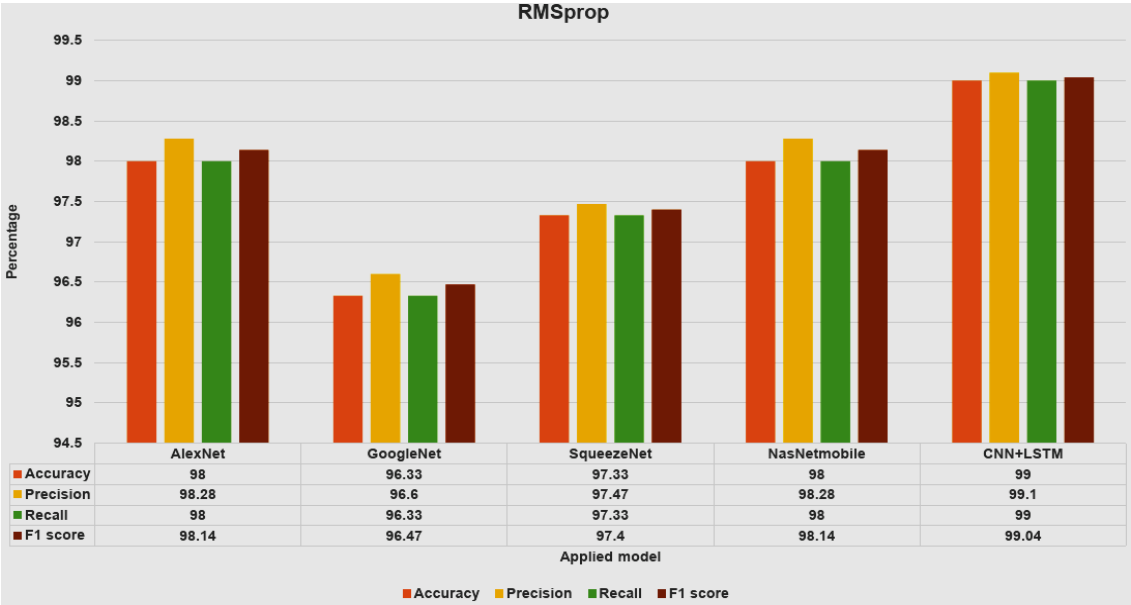


Fig 5.30 Comparison of various metrics for 5-fold cross-validation with different DL models for RMSprop Optimizer

From Figure 5.30, it is seen that the accuracy, precision, recall, and F1 scores are 99%, 99.1%, 99%, and 99.04%, respectively, for the 5-fold CNN-LSTM model. Among all other models for RMSprop optimizer, CNN-LSTM performs better.

The above figures 5.28 through 5.30, indicate the dominance of the proposed CNN-LSTM model in terms of Accuracy, Precision, Recall, and F1 score. It is noticed from above figures that within various optimizers, RMSprop is the best for CNN-LSTM model.

The result summary of all the DL models for different optimizers is presented in Table 5.2.

To deploy any DL model on an embedded platform the model size plays an important role. Table 5.3 shows the depth, size in MB, number of parameters, and size of the input image for all the models applied. In this thesis, an efficient CNN-LSTM model was developed that has the highest accuracy and is also the lightest among all the models, with a size of 2.13MB and 3664261 parameters.

Table 5.2 Summary of all the metrics for different Optimizers

Optimizer	Model	Average			
		Accuracy	Precision	Recall	F1 score
Adam	**CNN-LSTM**	**98.33**	**98.46**	**98.33**	**98.40**
	AlexNet	97	97.31	97	97.16
	GoogleNet	97	97.33	97	97.16
	SqeezeNet	96	96.45	96	96.22
	NasNet Mobile	98	98.24	98	98.12
Sgdm	**CNN-LSTM**	**98**	**98.2**	**98**	**98.1**
	AlexNet	97.66	97.86	97.66	97.77
	GoogleNet	95.67	95.93	95.67	95.8
	SqeezeNet	80.67	79.8	78.67	79.33
	NasNet Mobile	96	96.38	96	96.19
RMSprop	**CNN-LSTM**	**99**	**99.10**	**99**	**99.04**
	AlexNet	98	98.28	98	98.14
	GoogleNet	96.33	96.6	96.33	96.47
	SqeezeNet	97.33	97.47	97.33	97.40
	NasNet Mobile	98	98.28	98	98.14

Table 5.3: Specifications of DCNN models

Model	Depth	Size (MB)	Parameters	Input image size
AlexNet	8	227.17	59551208	227x227x3
GoogLeNet	22	26.70	6998552	224x224x3
SqueezeNet	18	4.71	1235496	227x227x3
NasNetMobile	27	19.02	4986040	224x224x3
CNN-LSTM	9	2.13	3664261	227x227x3

5.4. Discussion

Automated segregation of arecanuts is crucial in the food processing industry. Manual segregation takes a longer time, which leads to the degradation of nut quality. This further leads to low economic yield. Very few researchers have worked with de husked arecanut classification, as shown in Table 5.4.

Table 5. 4. Work carried out by researchers on de-husked arecanut classification

Researcher	Methodology	Accuracy (%)	Classification
Shedthi et al.	Logistic regression, k-NN, naive Bayes classifiers, and density features with SVM and ANN	98.8	Binary
K. Jyothi et. al	LBP for feature extraction and SVM for grading	77(LBP) 79(LTP)	Multi-class
Siddesha et al.	Wavelet, Gabor, LBP, GLDM, and GLCM for feature extraction and NN classifier	91.43	Multi-class
Bharadwaj et al.	Block-Wise LBP for feature extraction, Otsu thresholding for segmentation, and SVM classifiers	94	Multi-class

There is little work done for multi-class arecanut classification. Table 5.4 compares de-husked arecanut classification with various ML approaches done by a few researchers.

Shedthi B et al. [5] have classified healthy and diseased arecanuts by applying ANN with density feature and got an accuracy of 98.8%. K. Jyothi et. al [6] got an accuracy of 77% for LBP and 79% for the LTP feature extraction technique for multiclass classification. S. Siddesha et al. [7] have worked on the texture-based classification of arecanut into seven classes using various feature extraction techniques and attained an accuracy of 91.43% using the NN classifier. Bharadwaj et al. [8] worked on Arecanut Grading and Classification for four classes using SVM with feature selection and achieved an accuracy of 94%.

In case 1, we have done a binary classification of arecanut (healthy and diseased), and an accuracy of 100% was achieved using the MobileNet model. Among binary class classifications, this accuracy is reported to be the best.

In case 2, we have seen that the custom CNN model outperformed standard AlexNet, and we achieved an average validation accuracy of 97.33 % and 98.34 % for 5 and 10-fold, respectively.

In case 3, we have compared several popular DCNN architectures, including AlexNet, GoogLeNet, SqeezeNet, and NasNetMobile, for their performance in arecanut segregation. With CNN-LSTM and RMSprop optimizer, we attained an accuracy of 99.04%.

We have also investigated the impact of different optimizers, such as Adam, Sgdm, and RMSprop, on the performance of our models. Our experiments show that the RMS prop optimizer consistently outperforms the other optimizers, leading to better convergence and higher accuracy of 99 for 5-fold cross-validation. This finding highlights the importance of optimizer selection in training DL models for arecanut segregation.

Our results indicate that CNN-LSTM consistently outperforms the other architectures, achieving the highest accuracy, Precision, Recall, and F1 score among the tested models. This finding suggests that the CNN-LSTM architecture is well-suited for extracting features from arecanut images. A combination of CNN -LSTM networks can significantly improve the accuracy of arecanut segregation compared to other Models. Among the multiclass classifications, this accuracy is reported to be the best.

5.5 References

1. V. H. Kamble and M. P. Dale, “Machine learning approach for longitudinal face recognition of children,” in *Machine Learning for Biometrics*, Elsevier, 2022, pp. 1–27. doi: 10.1016/B978-0-323-85209-8.00011-0.
2. T.-T. Wong, “Performance evaluation of classification algorithms by k-fold and leave-one-out cross validation,” *Pattern Recognition*, vol. 48, no. 9, pp. 2839–2846, Sep. 2015, doi: 10.1016/j.patcog.2015.03.009.
3. Department of Computer Science and Informatics, University of Energy and Natural Resources, Sunyani, Ghana, I. K. Nti, O. Nyarko-Boateng, and J. Aning, “Performance of Machine Learning Algorithms with Different K Values in K-fold CrossValidation,” *IJITCS*, vol. 13, no. 6, pp. 61–71, Dec. 2021, doi: 10.5815/ijitcs.2021.06.05.
4. M. G. Lanjewar, J. S. Parab, and A. Y. Shaikh, “Development of framework by combining CNN with KNN to detect Alzheimer’s disease using MRI images,” *Multimed Tools Appl*, vol. 82, no. 8, pp. 12699–12717, Mar. 2023, doi: 10.1007/s11042-022-13935-4.
5. S. Shedthi B, M. Siddappa, S. Shetty, and V. Shetty, “Classification of arecanut using machine learning techniques,” *IJECE*, vol. 13, no. 2, p. 1914, Apr. 2023, doi: 10.11591/ijece.v13i2.pp1914-1921.
6. Jyothi K, Sindhu M Hegde, Sumedha K S, Sushma C R, and Thanushree D C, “Grading of Arecanut Using Machine Learning Techniques,” *IJSRCSEIT*, pp. 33–39, Jul. 2022, doi: 10.32628/CSEIT2283119.
7. S. Siddesha, S. K. Niranjan, and V. N. Manjunath Aradhya, “Texture based classification of arecanut,” in *2015 International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT)*, Davangere: IEEE, Oct. 2015, pp. 688–692. doi: 10.1109/ICATCCT.2015.7456971.
8. N. K. Bharadwaj and, R. Dinesh “Possible approaches to arecanut sorting / grading using computer vision: A brief review,” in *2017 International Conference on Computing, Communication and Automation (ICCCA)*, Greater Noida: IEEE, May 2017, pp. 1007–1014. doi: 10.1109/CCAA.2017.8229971.

CHAPTER 6

PROTOTYPE DEPLOYMENT AND VALIDATION

In the previous chapter, we have given a technique for classifying the arecanuts into five classes, and we have achieved an accuracy of 99% using Custom CNN-LSTM [1] and RMSprop [2] optimizer on a standalone system. Here, we have used distributed hardware consisting of a high-performance computer system with an Intel i5 processor having 32GB SDRAM and an NVIDIA GEFORCE RTX 3060 graphics card with 12GB DDR6 video memory. The reason for using such a system was to get the computational results quickly and more accurately. The images used for training and testing had a resolution of 5mega pixels and a total of 300 images. Deploying such an advanced system on a portable machine requires a custom-designed microcomputer and a high-speed camera with a resolution of 10MP. The cost outlay for such developmental work is enormous and is not affordable even for small farmers. However, after consultation with arecanut traders, it was found that the bottleneck of payment is the removal of bad quality arecanut from good variety. Once good varieties are segregated, the traders have a fair idea of the percentage of sub classes of good arecanuts, and they can estimate the amount to be paid to farmers on the spot. The farmer can then decide whether to sell the product to that trader or take it to a different trader. By knowing the weight of good nuts, the farmer also gets a fair idea of the amount due for him for that lot. With these constraints in mind and also the affordability of developmental costs, it was decided to design and develop a simplified instrumentation for two-class segregation. As discussed above, this portable instrumentation would bring down the cost of development and make it more affordable to even small farmers. The design can be described as below. All the arecanut received by a trader would be segregated into two-class i.e. good and bad. For such a classification, we can have a variety of mechanical designs; the most preferred normally is a conveyor belt model. However, such a model footprint is quite big and requires huge power resources. Though they are quite efficient, they require regular maintenance and often have breaks. The idea of our development was a portable model which could be carried by the farmers to the fields and could be run on low power UPS system without requiring a main power source. Also, as said earlier, the conveyor belt system, being huge in design, the basic cost of equipment can shoot up beyond the purchasing power of a farmer.

6.1 Design of Segregation Wheel:

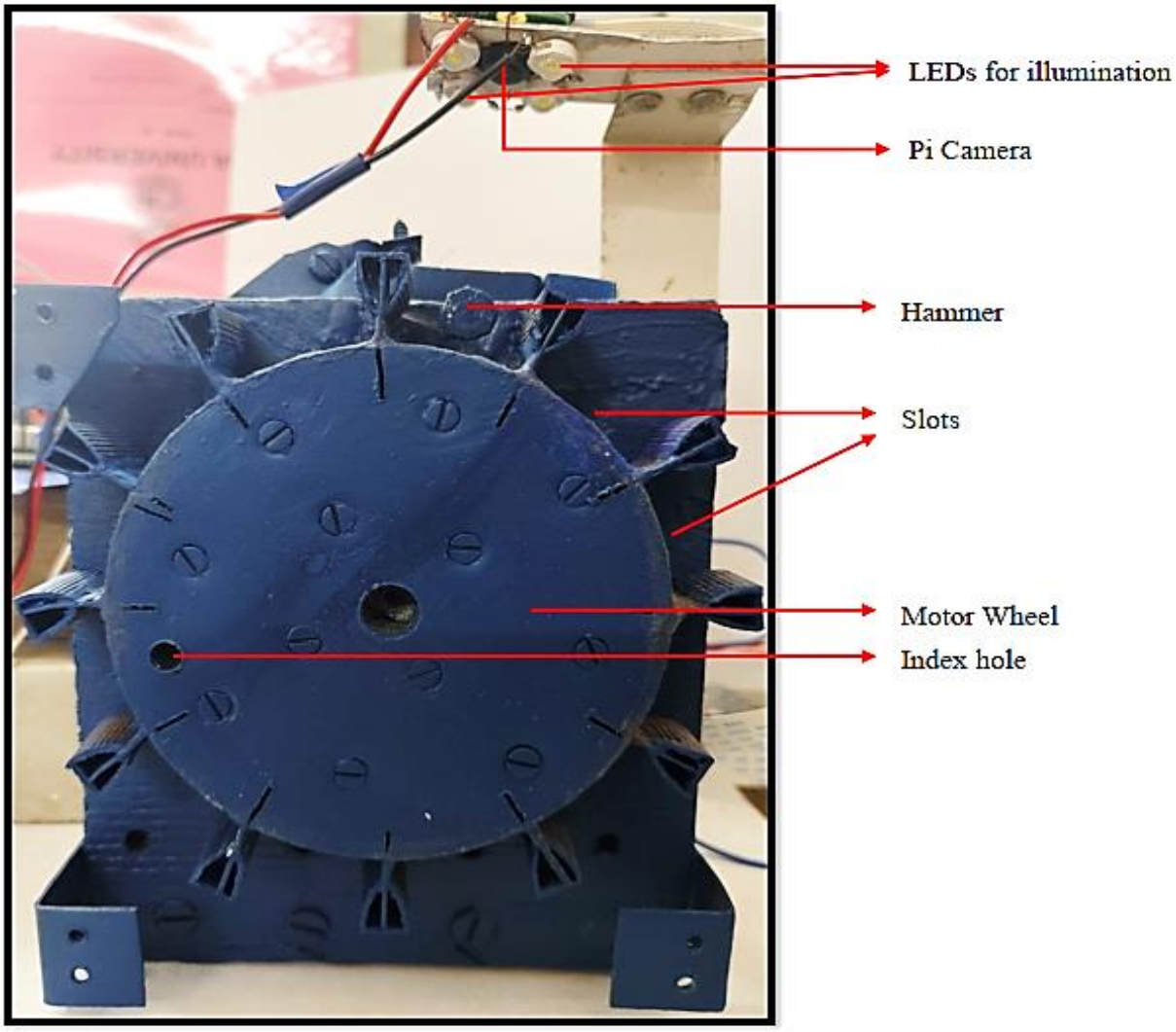


Fig 6.1: Front View of Bare segregation wheel with associated components

(a) Good Arecanut (b) **Bad Arecanut**

Fig 6.2: Images captured using the developed

After, evaluating other such designs, we have designed our own unique architecture, which consisted of a radially spoked wheel mounted on a stepper motor. Similar types of technology are being used for high-speed verification of labels in final products. Here, they use strobed illumination for short exposure [3]. The fig.6.1 gives the design of wheel, which has a total of twelve spokes. In other words, the outer side of the wheel has twelve compartments (Slots). In a radial arrangement, it is obvious that the outer side of the slot will have more width as compared to the inner side. Therefore, to make this compartment of equal dimensions on the inner and outer sides, a plastic object is glued to each of the spokes, as shown in the figure. Now, as you can see in Fig 6.2, the arecanut is accommodated comfortably in the slot centrally.

One can see from Fig 6.2 that Fig 6.2a has a good quality arecanut but is slightly smaller than a bad arecanut shown in Fig. 6.2b. We do not mean that bad arecanuts are always bigger in size, or for that matter, good arecanuts are smaller. We would like to convey here that we have designed the segregation wheel slots in such a way that the same slot can accommodate bigger or smaller arecanuts as long as two nuts do not come in a single slot. It may be seen from Fig 6.2 that the surrounding of the arecanut is quoted with specific hue blue colour. This colour and hue was chosen after many trials to get a proper image contour. After acquiring the image of the arecanut, during the preprocessing of the image, the software prepares a virtual mask for the arecanut, and only the information inside the mask is sent for processing purposes so that maximum information on the features of the arecanut is utilized for proper classification. This masking method gives high accuracy in the determination of the quality of the arecanut. When the background colour was painted full bright white, the white husk part of the arecanut used to get mingled with the background white, and the masking used to get disturbed, resulting in bad classification. Similarly, when the background was painted full black, the black fungus on the arecanut used to interfere with masking. After putting all the acquired

Fig. 6.3 Arecanut Size Segregator Drum Unit

images of arecanuts of different verity to the colour identification test, we found that a blue colour with a specific hue was not present in any of these images, and thus we could get a perfect virtual mask for preprocessing of arecanut images.

6.1.1 Wheel Dimensions

The wheel width of the segregation unit is 22mm, and a diameter of 6cm. The slot width from inside to inside is 22mm, thus, a square compartment is formed within which a single arecanut can be comfortably accommodated. The selection of 22x22mm. sqr is based on the physical verification of the sizes of arecanut brought by the farmers to the traders.

Arecanut Size Segregator: The sizes of arecanut seen in the market yard vary from 35mm to 15mm in diameter. The traders normally send them to a mechanical sorting unit, as shown in Fig 6.3[4], wherein they are passed through a long, slightly inclined horizontal drum having holes through the surface of the drum of different diameters varying from 15mm to 35mm. The drum is constantly agitated and turned at 10RPM. The arecanut are fed from left hand side square basket, which is then taken up by a lifter and poured into the funnel at the upper end of the drum and by the time they reach down, various sizes get accumulated in different gunny bags tied to the funnels placed along the drum at the appropriate location. This process is generally very fast and each of lot collected in the bags, whether they belong to a big category or small category or for that matter medium category, have the proportion of good or bad arecanut nearly same. This is due to the drying process adopted by the farmer after cutting the crops from the arecanut tree. Therefore, for determining the percentage of good or bad areca, it is sufficient to determine on medium size arecanut. We have observed that percentage of medium size arecanut of size varying from 18mm to 22mm is nearly 60% after the mechanical segregator.

Therefore, in our design, we have used slot sizes of 22mm so that the slot can accommodate an arecanut from size 18mm to 22mm comfortably. The arecanut is fed at position “A” using a funnel with a maximum outlet opening hole of 22mm diameter. The segregation wheel is rotated at 5 RPM.

6.2. Full Assembly of Segregator

The different views of the prototype mode developed for the arecanut segregation is given in Fig.6.4a, Fig.6.4b, and Fig.6.4c. The arecanuts to be segregated are pored in the funnel as shown in Fig 6.4a. The funnel output end is constantly agitated using an agitator constructed

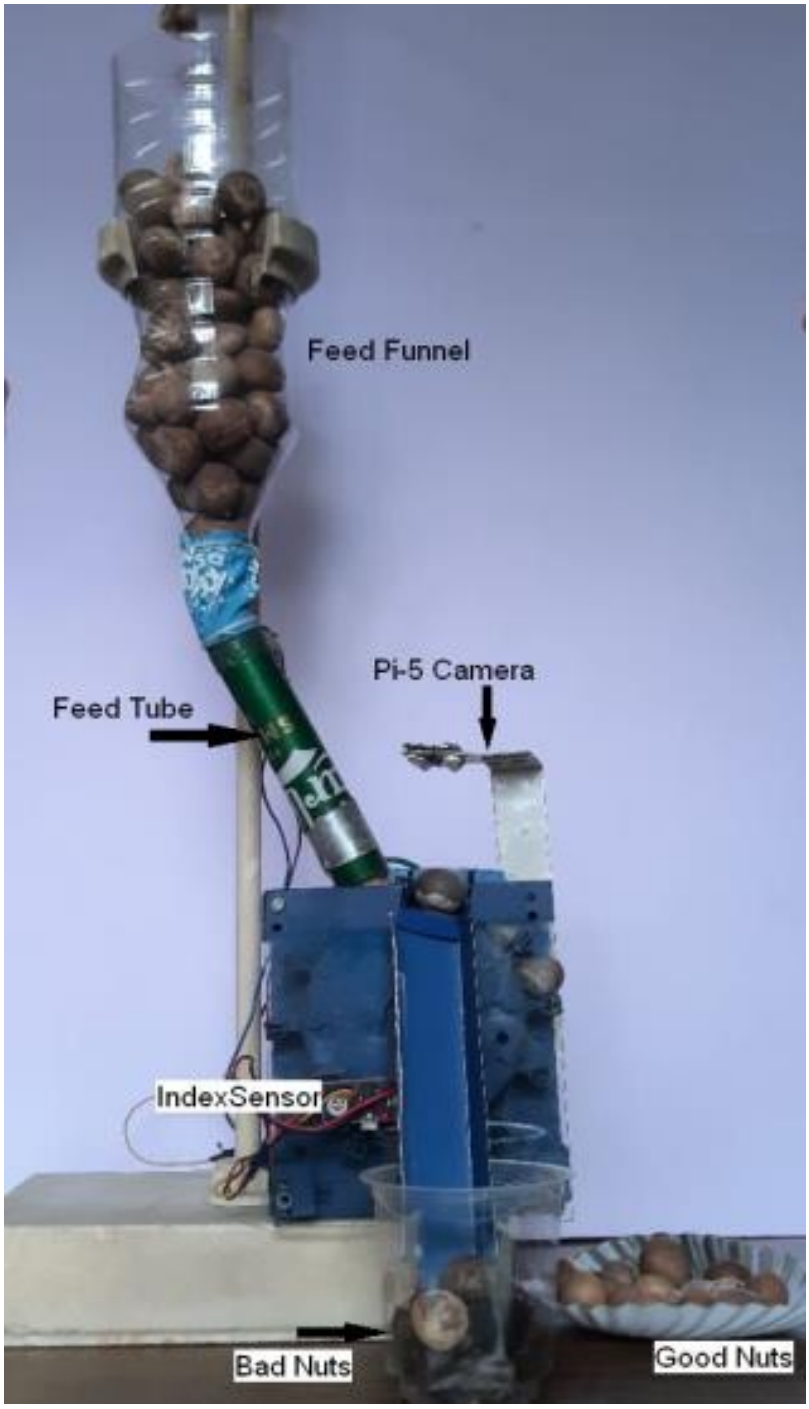


Fig 6.4a: Front View of Full Segregator System

with a motor loaded by an eccentric load, as shown in Fig. 6.4b. The rotation speed is limited by the speed of the camera used for capturing the image of the arecanut. Though the speed of the camera is pretty high (appr. 1ms for the image acquisition), the processing of the image to determine the quality of the arecanut in front of the camera takes 100ms approximately.

This is because the entire ML software is running on PI board with limited resources. As discussed earlier, we have used tensor flow 1.5 in Python environment, and thus it takes its own time resulting into overall 5RPM. Once the software decides the quality of the image acquired by the camera, is response signal is activated and given to the driver circuit of the hammer if the given arecanut is classified as bad. The hammer pushes out the arecanut from the slot and is collected in a bag tied to the funnel attached to the opposite side of the hammer, as shown in Fig. 6.5b. If the arecanut belongs to good quality, the classification software issues a command to the segregation wheel to move forward. The good arecanut moves forward and is collected in a bag attached to the funnel placed in the forward direction. It may be possible that there might

not be any arecanut loaded in the slot for any unknown reason, the software may take a long time to determine its decision.

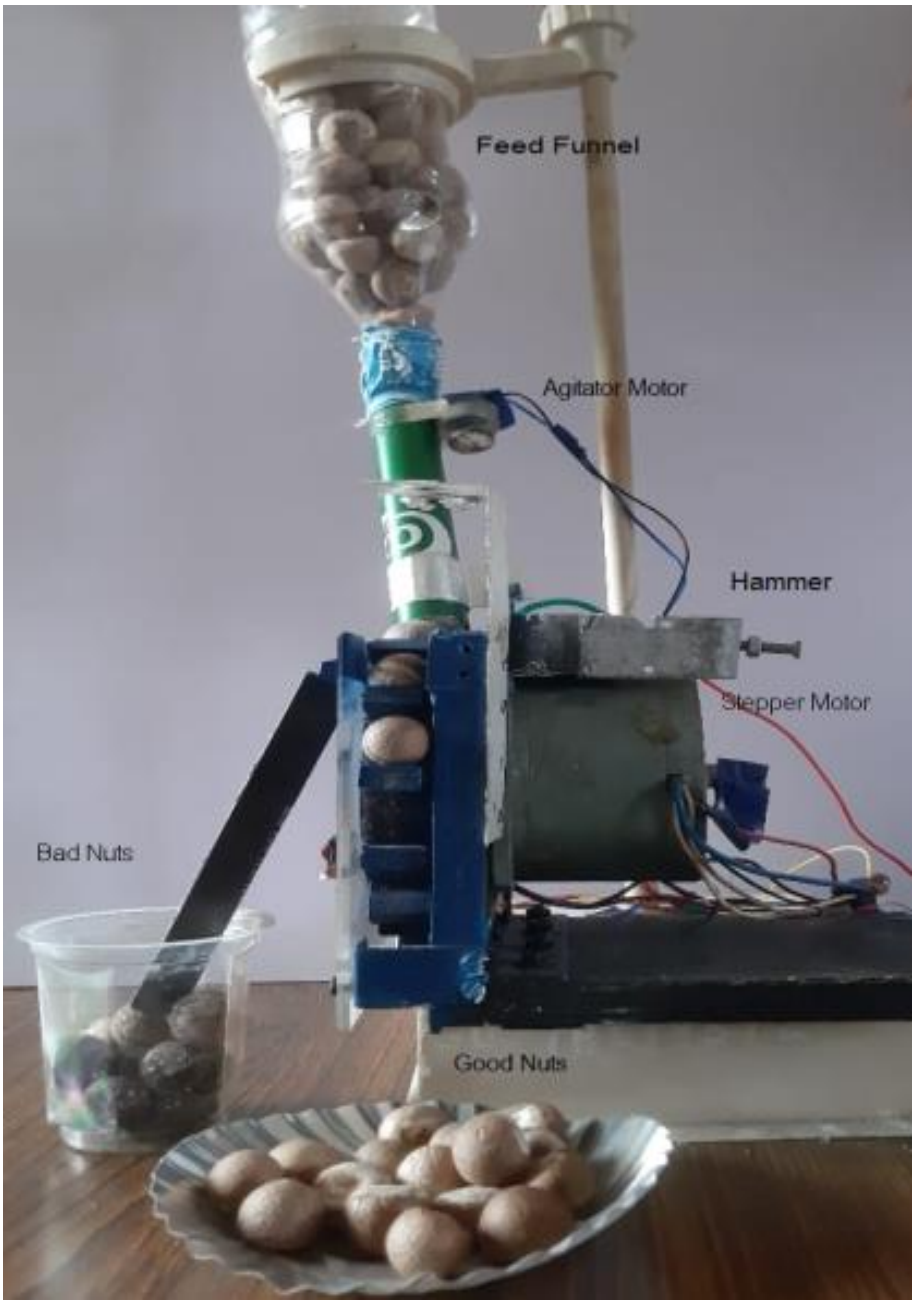


Fig 6.4b : Side View of Full Segregator System

To avoid such delay, a time-out subroutine is created in the classifier algorithms in such a way that, after waiting for 1ms, the wheel is moved to the next location without waiting for the decision from the software, and also the camera is reset for the next image of next arecanut on its way. The backside of the segregation wheel is covered by a wooden plank of size 8x10 cms. sqr, on which the stepper motor is mounted using four screws. The shaft of the stepper motor comes out from a hole drilled two centimeters above the center of the plank. The segregation wheel is firmly attached to the shaft using a shaft screw. as shown in Fig.6.5a A clearance of 0.1mm is maintained between the wooden plank and the segregation wheel for free movement of the wheel during the operation. The PI camera, along with object illumination LEDs are placed above 4cm from the wheel in such a way that the center of the arecanut placed in the slot comes in clear focus of the camera. All the LEDs must have equal illumination so that

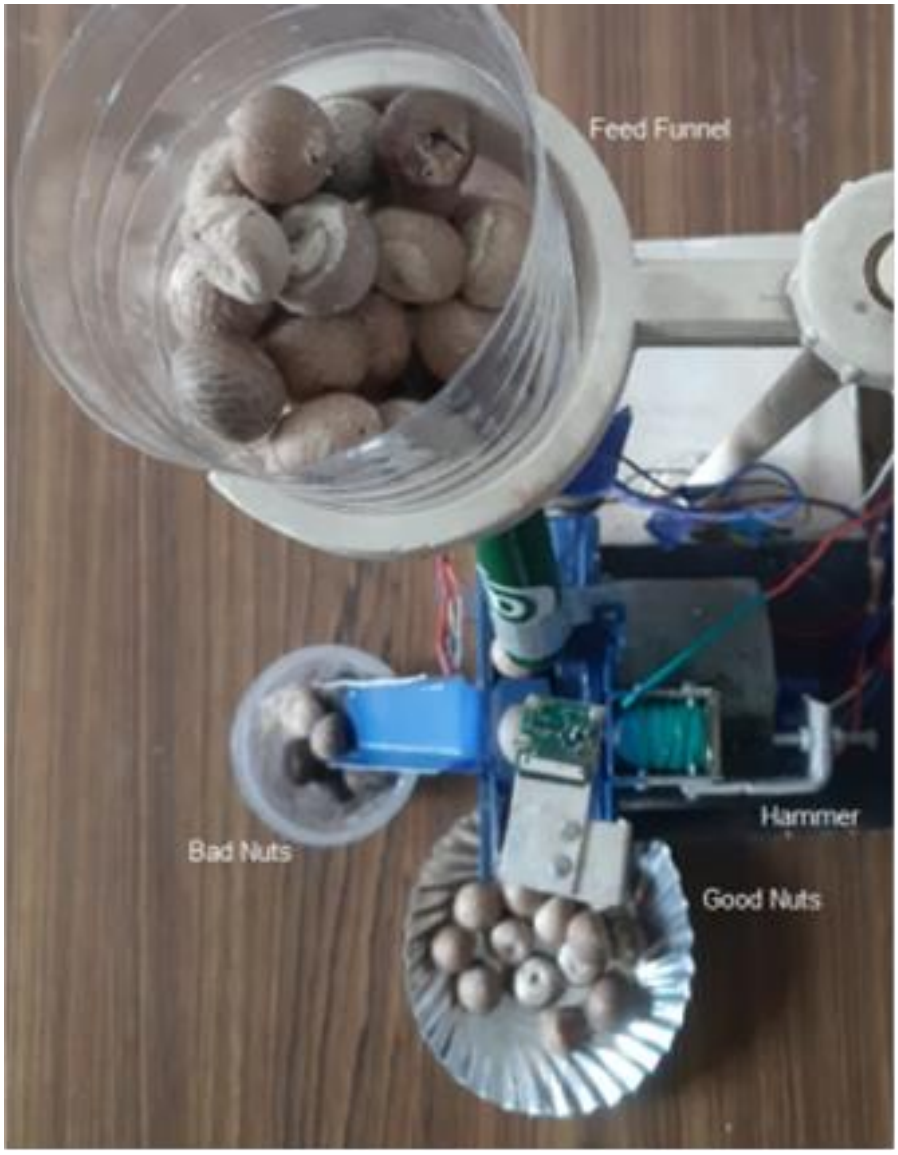


Fig 6.4c: Top View of Full Segregator System

differential shadow will not be formed on the arecanut. For this purpose, the same current is passed through all the LEDS, assuming they have identical characteristics (1 Watt High Power High-Intensity SMD LED in white were used from a single source)[5], as shown in fig 6.5c. It may also be seen in the fig.6.5b that the hammer for pushing out bad arecanut is also aligned with the slot centrally.

The hammer consists of a solenoid with an actuator working as a hammer and can be operated using a 24volts DC drive. The placement of the solenoid can be seen in Fig. 6.5b. The alignment of the arecanut and the hammer (The shaft actuator of the solenoid) can be seen in the same figure.

6.2.1 Index Hole Design:

The figure 6.1 also shows the location of an index hole. The index hole is a through a hole in the segregation wheel, the back side of which has a transmitter LED fixed to the supporting plank, and the front side of the index hole has a photodiode with a receiver circuit fixed on the front side acrylic sheet covering the segregation wheel, as shown in fig 6.5a. The circuit used for the sensor is shown in fig 6.6. The indexing unit functions as an initializer of the segregation wheel. It may be possible that, due to load or unforeseen conditions, the stepper

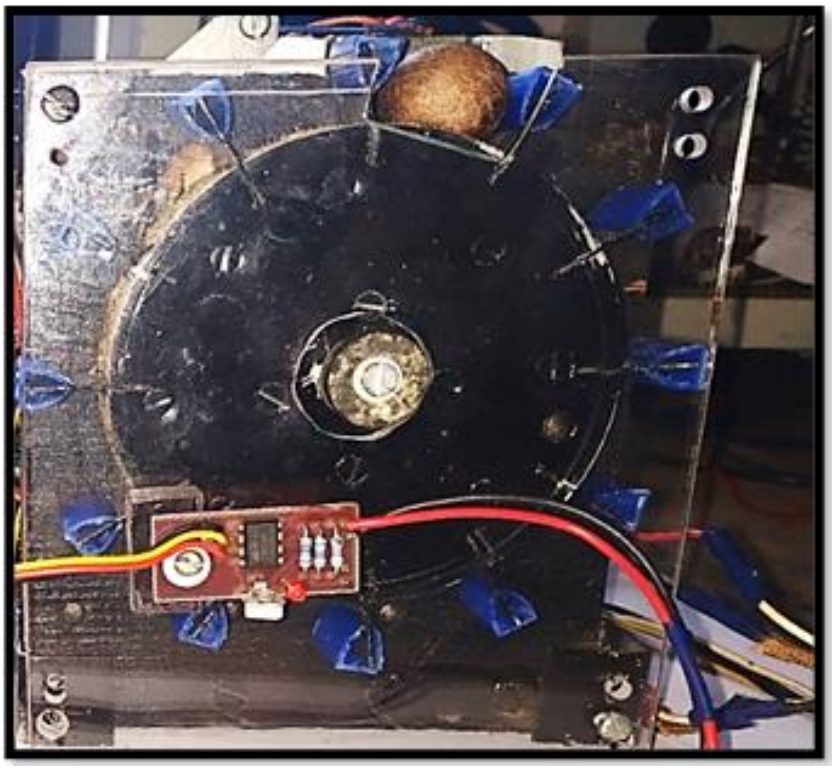


Fig.6.5a :Front View of Segrgation Wheel

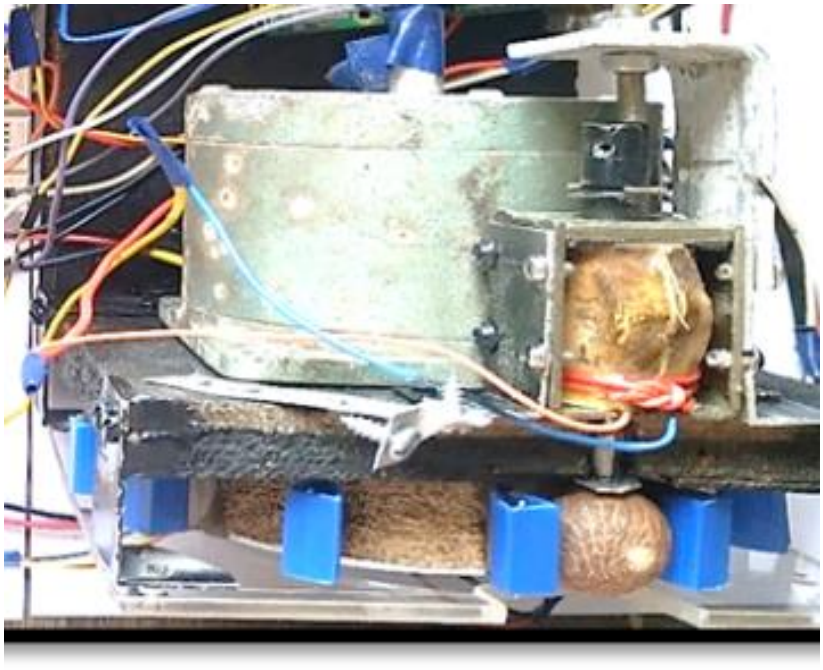


Fig.6.5b :Top View of Segregation Wheel

motor will lose one or two steps. If this is not corrected within a reasonable time, the steps may accumulate during the continuous process leading to non-predictable results. Therefore, up to 11 steps, the wheel is moved by a definite number of steps. After the 11th step, the program initiates continuous movement till the index receiver circuit (Rx) gets light from transmitter Tx on the other side of the wheel. Thus, the wheel is brought back to the starting position after every revolution, eliminating the missed step scenario. The LED at the output of the LM358 integrated circuit is a monitor to see if the indexing is functioning properly. Here, a general-purpose Op Amp is used in the comparator mode with a 50K trim pot, which decides the level of trigger depending upon the ambient light. When the light from Tx passes through the index hole, Rx becomes more conductive due to photo conducting nature of the photodiode. The voltage at pin2 starts rising as a result of this change, and the comparator changes its state, and the output of the Op-Amp jumps to the positive supply and sends the signal to the I/O port of the PI microcomputer,

Fig. 6.5c Placement of LEDs

indicating that the segregation wheel has reached the the start position of the operation. The stepper motor has a maximum of 200 steps/revolution. Since there are 12 slots in the segregation wheel, one slot corresponds to 200/12 steps (16.66 steps). As the steps per slots are non-integers, we have programmed the driving software to have a combination of 16 and 17 steps alternately to cover the one revolution in 200 steps. The stepper motor was procured from Sri-Jan Motors, Pune [6] and has 200 steps/revolution with a Torque of 3Kg.cm, and works on 24 Volts supply.

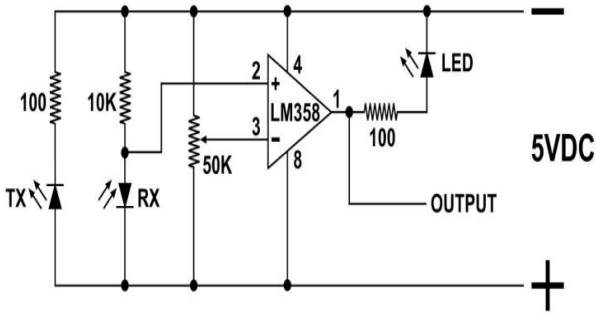


Fig.6.6 Sensor circuit for indexing

The choice of the motor was based on our requirements and its power needs and motor configuration. Lesser torque than 3kg. cm leads to frequently missed steps.

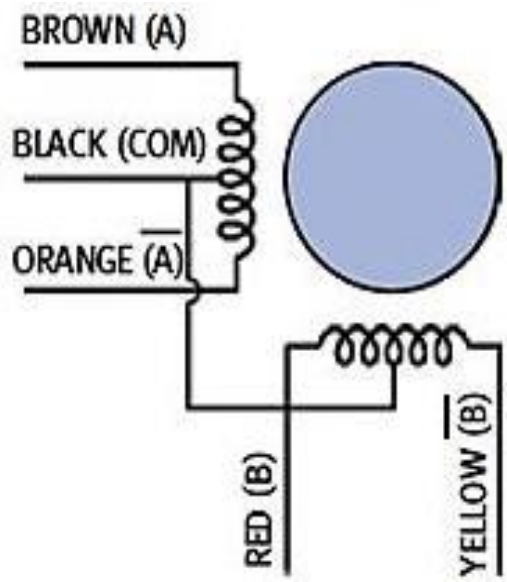


Fig 6.7: Wire Stepper Motor Diagram

6.2.2 The Wiring of Stepper Motor

The motor supported 4-phase operation with full step for higher torque. Step resolution was not an important criterion, as one-step miss alignment was not much of an issue for the operation. There are basically four ways the six terminal 4-phase stepper motor can be configured, namely a) Unipolar Half-Coil and b). Bipolar-Series c) Bipolar-Parallel and d) Bipolar [7]. The unipolar drivers are more cost-effective; however, bipolar drivers offer more flexibility and allow different ways to connect to six-wire stepper motors. The wiring of the Unipolar 4 phase stepper motor is given in Fig. 6.8. The term unipolar is used here to specify that current in a given coil is moving in only

one direction, and a single DC supply is sufficient to drive the motor. The circuit shown here is also called a half coil because, at a given time, only 50% of the coils are energized while the remaining coils go into idle mode. For bipolar-series wiring, as shown in Fig 6.9, we use the full coil (winding). By using the entire winding, the motor will be able to generate more torque compared to unipolar. However, the inductance increases by fourfold, so the torque of the motor drops off rapidly at upper speeds. Also, during holding time, all the coils will draw current simultaneously, and the power supply drive current is double. Secondly, four transistors are involved in switching; failure rates are higher. Bipolar-parallel wiring is shown in Fig 6.10. This configuration has better possible speed and torque characteristics.

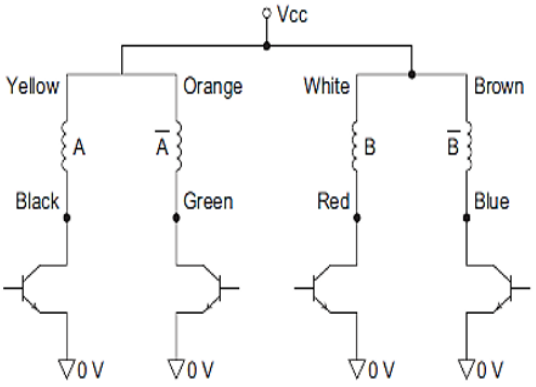


Fig 6.8: Wiring diagram of 4 phase motor

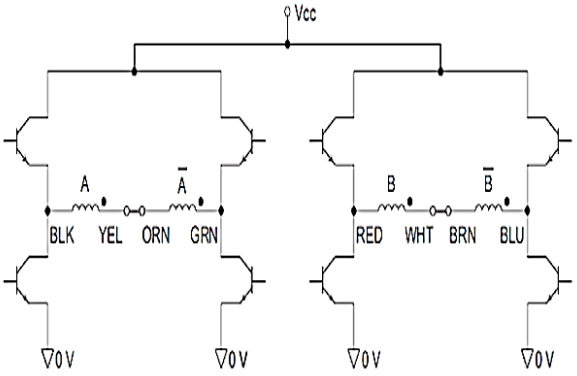


Fig 6.9: Wiring of 4 phase Bipolar-Series motor

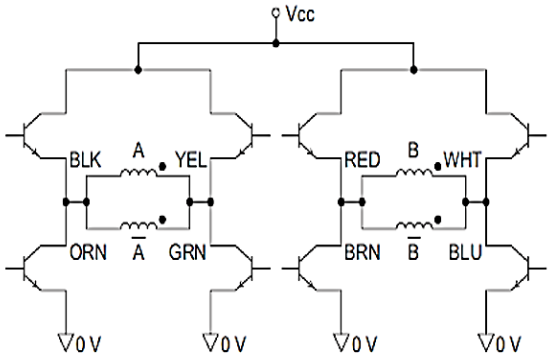


Fig 6.10: Wiring of 4 phase Bipolar-Parallel

This mode of wiring configuration uses all coils, so the torque is increased in comparison to unipolar. However, during holding time, all the coils will draw current simultaneously, and the power supply drive current is double. Secondly, four transistors are involved in switching, failure rates are higher

We used a 6-wire unipolar half-coil scheme for our developmental work. The choice was purely based on the simplicity of the circuit and the higher mean time between failures. It was often found that coil drive transistor blows quite often in inductive load operation due to high peak inverse voltage during drive current cut off from the coils. A full-step mode produces the highest level of torque with low vibration, but the limitation is that it provides the lowest resolution among all the drive circuits. In the above operation mode, phases A and B of the steeper motor are engaged simultaneously, thereby

providing a higher force relative to the other drive modes while keeping the resolution of this method constant. Every step in full-step mode rotates the stepper motor in a calculated angular distance. To get more torque, we used a full-step drive sequence, as shown in Fig. 6.11.

Step Sequence

Motor Phase	1	2	3	4
A	1	0	0	1
B	1	1	0	0
$\overline{A}$	0	1	1	0
$\overline{B}$	0	0	1	1

Fig.6.11 full step drive sequence of stepper motor

6.3. Drive Circuit for System

A typical drive circuit for the stepper motor is shown in Figure 6.12. Such four circuits are needed to drive the motor. The transistor Q1(IRF 740) is a reverse voltage-protected MOSFET device used in digital drive circuits having inductive loads. It has a reverse breakdown voltage of 400 volts and a current rating of 10 amps [8]. It also has a low on-time resistance of 0.48 milliohms, and thus, power dissipation during on-time is quite less, which makes it ideally suited for our developmental model. Capacitor C1 and Diode D1 further enhance the durability of the MOSFET during high voltage surges. The drain is connected to the stepper motor coil, as shown in Fig 6.8. The MOSFET Q1 is driven by the optocoupler U1(4N35). The resistance R1 is connected to the output of the PI microcomputer. The opto coupler is essential in such drive circuits as voltage surges during the stepper motor coil-switching can harm the output port of the PI microcomputer. The complete circuit diagram of the arecanut segregator drive circuit diagram is given in Fig. 6.13. A 5-pin connector designated as “MOTOR” goes to a stepper motor, which also includes a 24-volt drive voltage. The various voltages required for different operations are brought out through connectors PWR1-24V, PWR2-12V, and PWR3-5V. The agitator connector gives 5V to the agitator motor for smooth feeding of the arecanut to the segregator wheel. The Solenoid connector used for the hammering device provides 24 volts, while CAM_PWR provides 5V to the PI microcomputer. MCU_INDEXING and INDEXING are the connector to receive signal from Rx and provide drive to Tx. MCU_LED is a connector for LED indicators that provides valuable information on the status of classification. MCU_MOTOR is the I/O port of the PI microcomputer unit. LEDs D6-D9 are monitors of MOSFETs to see if they are blown during the operation. Figure 6.14: shows the Component layout Driver Circuit of the Arecanut segregation unit.

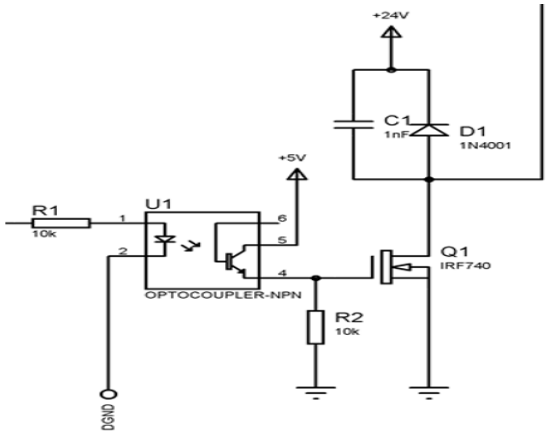


Fig.6.12 Drive circuit for one coil

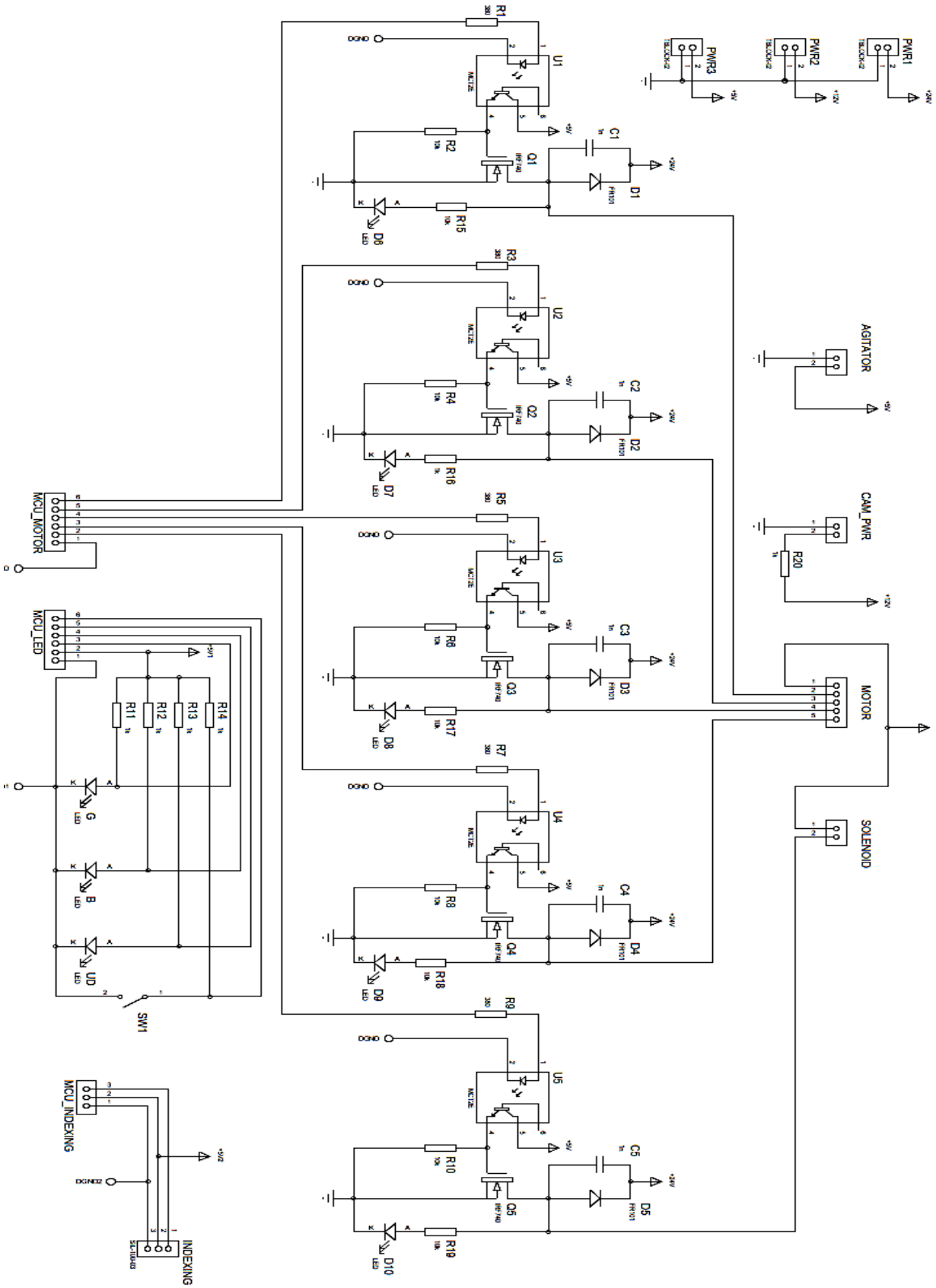


Fig 6.13: Circuit schematic of Driver Circuit of the Arecanut segregation unit

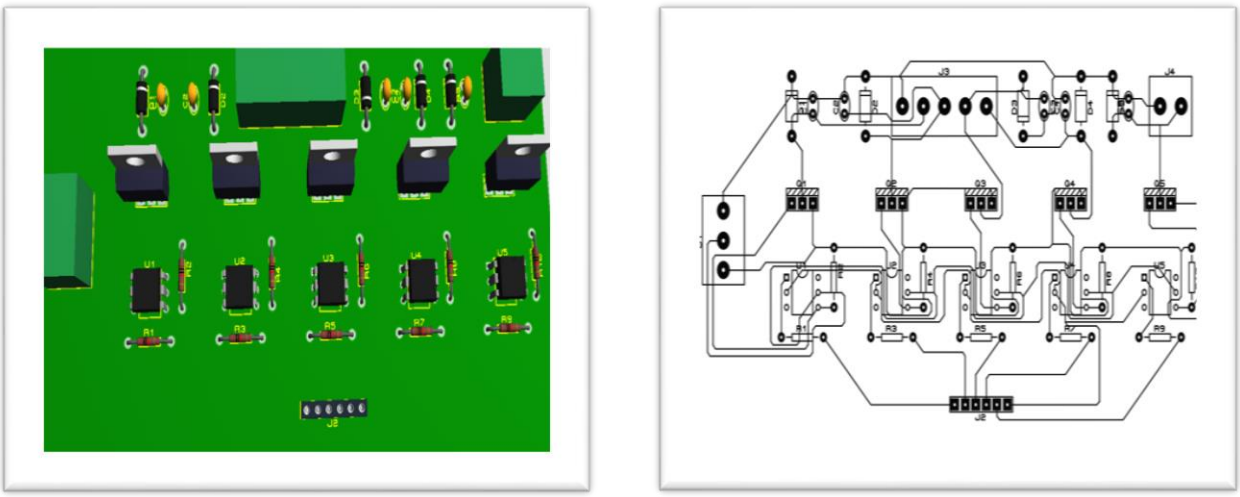


Figure 6.14: Component layout Driver Circuit of the Arecanut segregation unit

6.4 Design of Power Supply

The power supply required for the drive circuit was designed using pulse width modulator UC3843, as shown in Figure 6.15. The UC3843 is a fixed frequency-mode PWM controller, it is specially designed for Off-Line and DC to DC converter applications with minimum external components[9]. These integrated circuits feature a trimmed oscillator for precise duty cycle control, a temperature compensated reference, a high gain error amplifier, a current sensing comparator, and a high current totem pole output for driving a Power MOSFET SPP11N60. The MOSFET used has 600V peak reverse breakdown, 20Amp forward current, and 0.18miliohms as On-Time resistance[10]. It also has a built-in reverse protection diode. The current sense input coming from voltage developed across R6(0.5ohms) makes this module suitable for the protection of the motor connected at the output. If the MOSFET fails(often leading to a short circuit), the current in R6 resistance will increase beyond the threshold of the PWM module, and circuit operation stops, thereby protecting the valuable load at the output (Stepper motor).

6.4.1 Input surge and SMPS fault protection

This section consists of two components, L1 and C10. L1 is an inductive filter and removes spikes in the input AC supply. Similarly, C10 bypasses spurious overshoots. The input supply of 230VAC is converted by using four 1N4007 Diodes that make up a full bridge rectifier, a 1N4007 has a PIV of 1000V and 1A rated rectifier diode. The filtering is done by a capacitor of value 100uF/400V. The circuit can run even for a 22uF capacitor. However, the

stepper motor has four coils that need 1.7 A per coil for operation, and therefore, a 100uF 400V capacitor was chosen for a 100W circuit load.

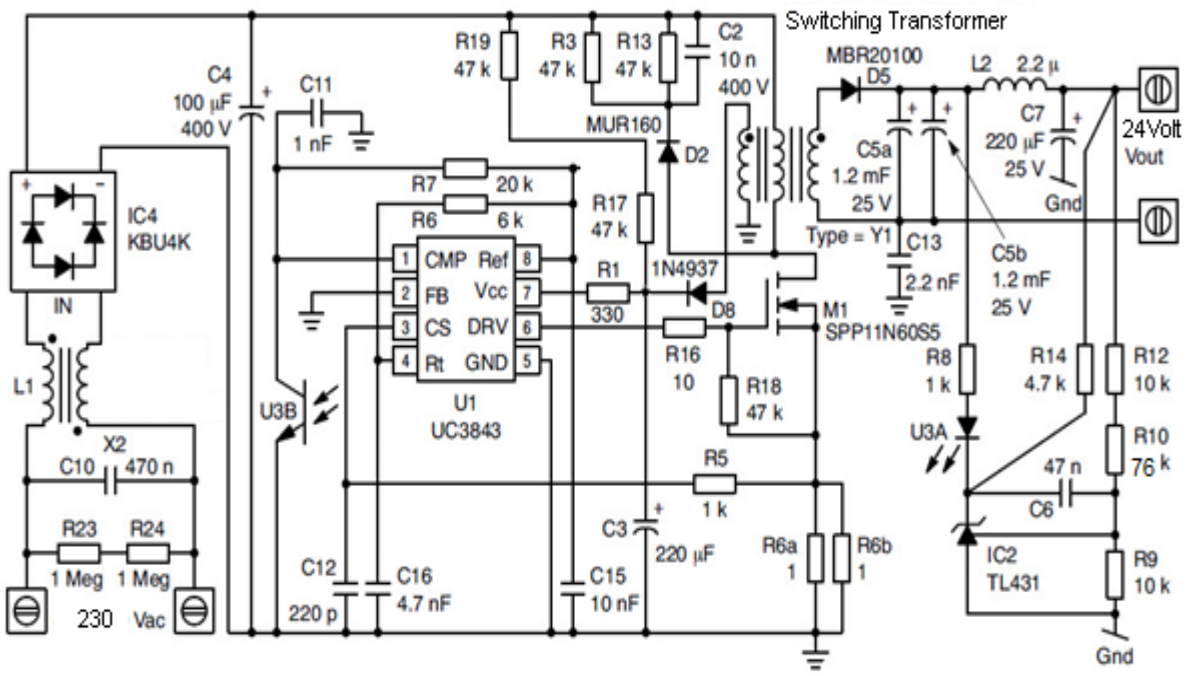


Fig 6.15 Circuit of 5Amp- 24 volts power supply for segregation unit

6.4.2 Driver Circuitry or Switching Circuit

For this design, the UC3843, alongside the SPP11N60 MOSFET, forms the main driver circuit. For the initial start-up of UC3843, some power (14Volts) is needed to start the operation, and the two resistors, R17 and R19, used are called the Start-up Resistors. These start-up resistors provide the initial start-up current to switch on the circuit, and when the circuit is switching, it is on a self-generated supply voltage of 11 volts produced by the auxiliary winding of switching transformer T1 and rectified by the D8 and C3 filter circuit. Reference voltage generated at pin8 is provided as supply to opto coupler receiver (U3B) section through R7 resistor.C15 capacitor is filter for reference voltage.

Clamp Circuit: The transformer is basically an inductor connected across the power MOSFET. Therefore, when the transformer goes to the off state, it creates a huge voltage spike. If this spike is not eliminated correctly, it could easily burn the MOSFET, which is why a clamp circuit becomes necessary. Thus, the C2, R3, R13, and D2 make up the clamp circuit. R18 ties the MOSFET gate to the ground when there is no signal.

6.4.3 Secondary Rectifier and Filter

We need to convert the output of the transformer T1 to DC before we can connect our application circuits. A D5(SR360) Schottky rectifier diode is used as the output current is 4A, and D5 is a 6A 60V rated Schottky diode. D5, C5, L2, and C7 take care of providing clean DC output.

Oscillator Frequency Selection: The frequency of the UC3843 IC can be adjusted according to needs, for our case, the Frequency of the IC is set to 80KHz with the help of the R6 Resistor and C16 capacitor. And an additional C12 capacitor is used to filter the power.

Fig. 6.16 Prototype Power Supply Board

Under Voltage Lock Out (UVLO)**:** When the input voltage of the power supply drops below the configured voltage (the coils of the stepper motor can get overheated due to more current, a common phenomenon in DC/AC motors), the UVLO detection voltage kicks in, the UVLO sets the internal circuit to a semi-standby state to prevent any damage to connected load at the output. When the power supply voltage rises and becomes higher than the UVLO release voltage, and the normal operation continues During Under-Voltage Lock-Out, the output driver is biased to a high impedance state. Pin 6 should be shunted to the ground with a bleeder resistor to prevent activating the power switch with an output leakage current.

6.4.4 Output Voltage Stability

This is done using variable voltage reference module IC2 (TL431). IC2 is a low-power device used for voltage regulation like Zenner diode. If the voltage at the reference pin, which is connected to the junction of the R9 and R10 resistor, goes above 2.5Volts, then the device starts conducting, thereby the LED of the U3A optocoupler switches on through the current limiting resistor R8(1k). This, in turn, switches on the receiver section transistor of the optocoupler U3B.

Higher conduction of the said transistor reduces the duty cycle of PWM module UC3843, and accordingly, the time of the MOSFET M1 gets reduced, and the output voltage gets reduced. This reduced voltage is fed back to TL431 through a resistor network comprising R9, R10, and R12. To maintain the output voltage at 24 volts, we need resistors of values 10k, 76k, and 10k, respectively. R14 is used as a constant bias for the TL431 device, and C6 removes the noise between the cathode and reference pin.

6.5 Microcomputer for Arecanut segregator

At the start of the development, a survey was done to choose an appropriate platform for running the classification algorithm and also drive the mechanical system very efficiently. The microcomputer needs good interface support to interface a camera with a minimum resolution of 5 megapixels. The other concern for choosing a microcomputer was the low power requirement and a good number of I/O ports for system status indication. The most popular system used by the developer is Arduino Uno [11] board because of its simplicity and low cost. However, this board does not have a good interface for the camera and does not support a Python environment. Moreover, this board has only 256KB of memory and is an 8-bit computer. The other popular board is the Altera DE0-Nano board [12]. This is a compact-sized FPGA development platform suited mostly for preliminary circuit designs such as "portable" projects. The board is most suited in the simplest possible implementation with the Cyclone IV type target device with up to 22,320 Logic gates. The board has a collection of interfaces, including two external GPIO headers to extend designs beyond the DE0-Nano board. On-board memory devices include SDRAM and EEPROM for larger data storage and buffering. DE0-Nano provides developers with three power options like: a USB mini port, a 2-pin external power jack, and two DC 5V connector pins. In spite of all the good things about this board, support for the camera requires large resources of the board, and all peripheral ports required for the segregation unit need to be burned using logic gates. Also, we found that it is very difficult to install a Python environment for running tensor flow.

6.5.1 Choice of Raspberry Pi 5 for System Design

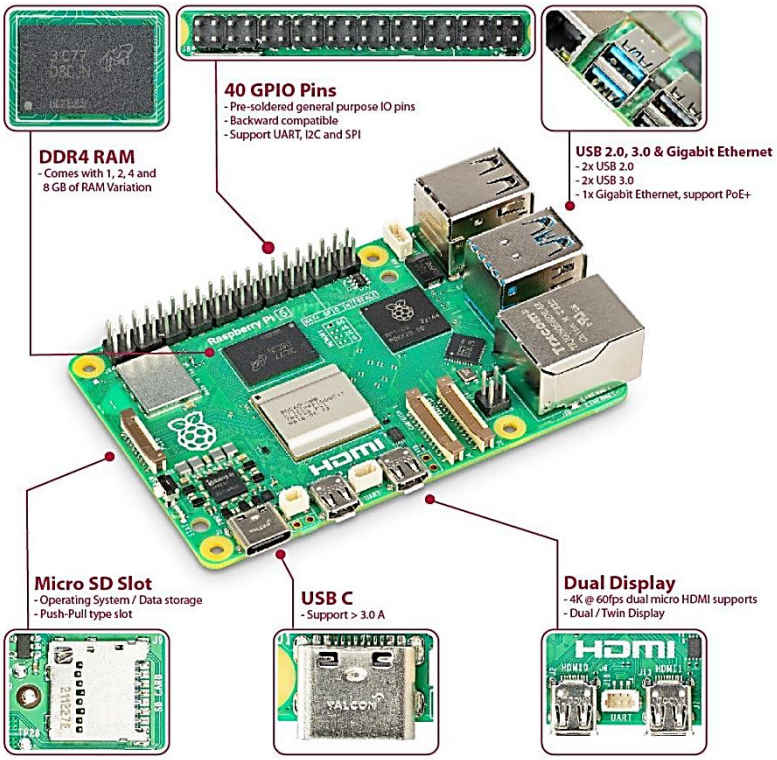
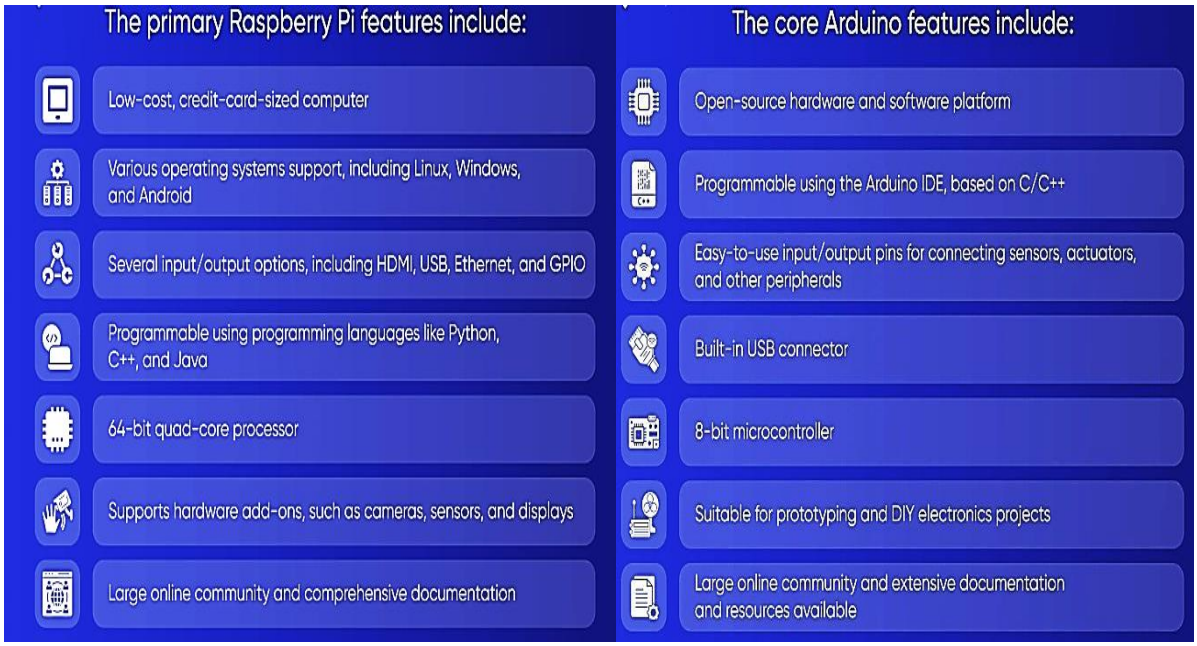


Fig 6.17. Functional description of Raspberry

Another popular development board is the Raspberry Pi 5 B[13] single-board computer with a Broadcom BCM2835 processor working at 1.5GHz and 8GB RAM. Raspberry Pi is a microcomputer board developed by the Raspberry Company in the UK to promote the learning of basic computer science in educational institutions and developing countries. Raspberry Pi has become a very useful microcomputer platform even outside of the education field. Science and technology, IT, and even small industries have been using Raspberry Pi-5 in their system. The Raspberry Pi board is powered using an independent adapter. Raspberry Pi boards act like typical standard PCs but in a more compact form factor. It has a Broadcom BCM2711 CPU, VideoCore VII GPU, 8GB RAM, HDMI and USB Ports, wireless and wired connectivity, Camera interfaces, and a microSD slot. Besides, it also has General Purpose Input/Output (GPIO) Pins that you can use to connect with other components, turning your single-board computer into prototype systems, as shown in Fig 6.17. This opens up more possibilities for what can be done with the Raspberry Pi board. A comparison between Raspberry Pi 5 and Arduino Board is given in Table 6.1. Equipped with ARM Cortex-A76 CPU clocked at 2.4GHz and VideoCore VII GPU clocked at 800MHz. Raspberry Pi 5 is a better option for your project that requires high-speed processing power like DL programs or intensive graphical rendering. Raspberry Pi 5 is designed to accommodate 8GB, 4GB, 2GB, or 1GB of LPDDR4X-4267 RAM options. Table 6.2 gives the functional description of Raspberry Pi 5. It comes with two USB 3.0, which is much faster than USB 2.0. which is perfect for any high-speed device connection like external SSD or USB drives. Raspberry Pi 5 can support up to 5V at 5A power supply through the USB type C connection.

Table 6. 1: Comparison between Raspberry Pi 5 and Arduino Uno microcomputers



Two USB 2.0 are also included for you to use with simple peripherals like keyboards and mice. with the RP1 IO, there is a "Southbridge" on Raspberry Pi 5. The RP1 has two high-speed USB 3 controllers, which are dedicated to a USB 3.0 port and another USB 2.0 port, respectively. The USB 3.0 port on this platform supports a maximum of 5Gbps data transfer. The 2 x 4 lane MIPI port supports both Display (DSI) and Camera (CSI) at the time, at either port. On Raspberry Pi 5, one could also have 2 cameras or 2 DSI displays at the same time, giving good performance. However, we need an adapter FPC/FFC for the camera and display; the FFC that comes with most of the camera module or display cannot fit into the new compact 22-way CSI/DSI socket. The Raspberry Pi 5 has a Power Management IC (PMIC). Besides supporting higher power management, it also comes with a built-in RTC (Real Time Clock). To utilize the RTC, it needs to be powered with an external battery. Raspberry Pi 5 comes with a dedicated battery port. With the battery, the Real Time Clock (RTC) will keep the clock counting when the Raspberry Pi 5 is switched off, thus keeping all time-related projects intact and reliable. The platform is provided with a built-in UART port.

Table 6.2: All specifications of Raspberry Pi 5

Features/Specs	Raspberry Pi 5
Announcement Date	28th September 2023
SoC Type (Processor)	Broadcom BCM2712
Core Type	Cortex-A76 64-bit (ARMv8)
No. of Cores	Quad-Core
CPU Clock	2.4 GHz
GPU	VideoCore VII
GPU Clock	800 MHz
Multimedia/Graphic	4Kp60 HEVC (H.265) Hardware Decode H.264 decode - from CPU H.264 encode - from CPU OpenGL ES 3.1, Vulkan 1.3
OS storage	microSD (support higher speed - SDR104) NVMe SSD (M.2 HAT via PCIe socket) NVMe or SATA SSD (Via USB3.0)
RAM	LPDDR4X-4627: 8GB, 4GB, 2GB* and 1GB*
Ethernet (RJ45)	Gigabit Ethernet
USB Port	2x USB 3.0 (Simultaneous 5Gbps) 2x USB 2.0
HDMI	2x micro HDMI Support True 2x 4Kp60
WiFi	802.11 b/g/n/ac /n/ac
Bluetooth	5.0 + BLE (Shielded)
Antenna	PCB Antenna
GPIO	40-Pin Headers
Dedicated Port	UART RTC External Battery Cooler Fan
Real Time Clock (RTC)	Yes, built-in
Power Button	Yes
Camera Port (CSI)	2x 4 lane MIPI camera or display transceivers. Support 2x Camera or 2x Display or 1x Camera + 1x Display
Display Port (DSI)	
PCIe Interface	1x PCIe Gen2.0
Operating System	Raspberry Pi OS Bookworm (> 28 September 2023)
Dimension	85mm x 56mm
Power Input	Full Power: USB-C PD 5V@5A PSU Normal Power: USB-C 5V (upto 3A) 5V via GPIO header (upto 3A) Power over Ethernet, requires New PoE+ HAT

One can use this for any UART communication device. This UART port also supports debug mode. The UART debug mode is always enabled by default so you do not need to worry about needing to enable it in the terminal under the configuration file. This UART port has its own dedicated pin and has independent UART pins on the 40-pin GPIO. The port is compatible with the Raspberry Pi -5 Debug Probe. Raspberry Pi 5 uses smaller CSI/ DSI connectors than the Raspberry Pi 4; we use an adapter cable or a module for the Raspberry Pi, as shown in Fig.6.17. Fitting the camera cable is pretty much the same as before turn off your Raspberry Pi, very gently pull the clip up, slot the cable in as far as it will go, and then push the clip down again Raspberry Pi 5 comes with a cooling fan port, Raspberry Pi 5 comes with a dedicated Fan connector. The fan port will provide power PWM control and Tachometer feedback from the cooling fan. The fan control and feedback pins are dedicated and independent from the 40-pin GPIO**.** It supports high-speed devices such as NVMe SSD, Network Card, or Graphic card.

6.5.2 The Camera Interface

The Raspberry Pi Camera uses the CSI interface to connect to the Raspberry Pi. The Camera features a 5MP (2592x1944 pixels) Omnivision 5647 sensor in a fixed focus module. The module attaches to the Pi board by a 15-pin Ribbon Cable, to the 15-pin MIPI Camera Serial Interface (CSI), which is designed for interfacing with cameras. The CSI bus is capable of very

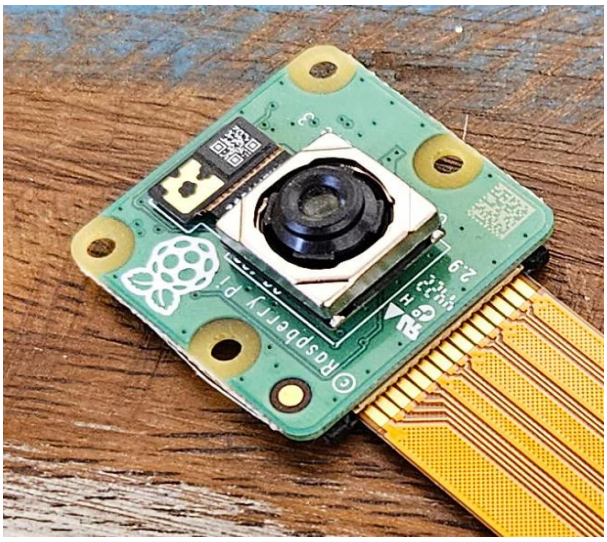


Fig 6.18: 5MP (2592x1944 pixels) camera

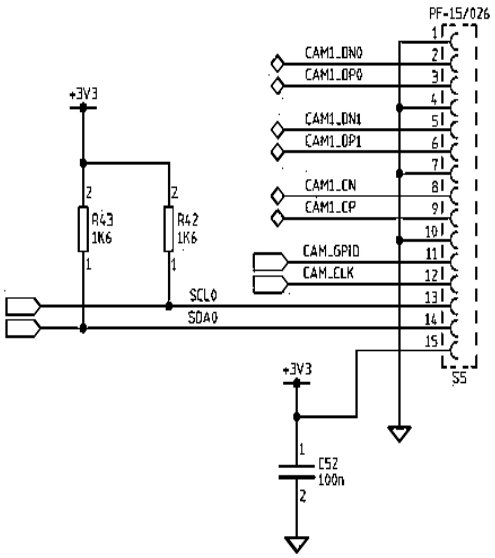


Fig.6.19 Schematic of the camera connector

high data speeds, and it exclusively carries pixel information to the BCM2835 processor. The camera board, mounted on the setup, has a dimension of 25mm x 20mm and is lightweight, making it perfect for mobile and other applications where small size and weight are important. The camera sensor has a resolution of 5 megapixels and has a fixed focus lens onboard, which can be manually focused for any given distance. The camera resolution for still images is 2592 x 1944 pixels, and it also supports different modes of video capturing, such as 1080p for 30fps, 720p for 60fps, and 480p for 90fps. However, we have only used the still image mode. The camera specifications are outlined below.

- Fully Compatible with Both the Model A and Model B Raspberry Pi
- 5MP Omnivision 5647 Camera Module
- Still Picture Resolution: 2592 x 1944
- Video: Supports 1080p @ 30fps, 720p @ 60fps and 640x480p 60/90 Recording
- 15-pin MIPI Camera Serial Interface - Plugs Directly into the Raspberry Pi Board
- Size: 20 x 25 x 9mm
- Weight 3g

The Raspberry Pi Global Shutter Camera has shorter exposure times, down to 30µs, and can collect enough light. This makes it useful for quick data acquisition. The flex cable inserts into the connector labelled CAMERA on the Raspberry Pi, which is located between the Ethernet and HDMI ports. The cable must be inserted with the silver contacts facing the HDMI port. The flex cable should be inserted firmly into the connector, with care must be taken so as not to bend the cable at high acute angles. Connector is closed by pushing the top part of the connector towards the HDMI connector, while holding the flex cable firmly in place. The connection diagram of camera interface is given in Fig 6.19. The principle is the same for all Raspberry Pi boards with a camera connector, though the Raspberry Pi 5 and all Raspberry Pi models require a different camera cable.

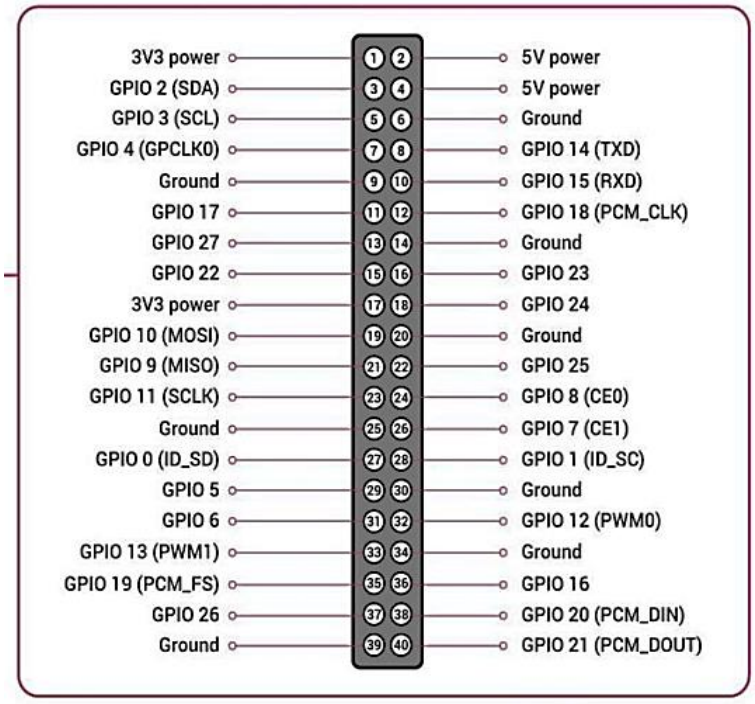


Fig.6.21 Raspberry Pi 40-pin GPIO connector

An Arduino board is an open-source microcontroller device used for interactive electronic projects. It comprises input/output pins, voltage regulators, USB interfaces, crystal oscillators, and other microcontroller components on a single circuit board. An Arduino controller can be used to read a variety of inputs. For instance, a sensor's light or a user pressing a button. Once read, it produces an output, turning on or off a lamp or starting an engine. However, there is no direct support for camera in the Arduino microcomputer and is only 8bit device. Tensorflow is difficult to work with as there is no Python support. Arduino has 256KB of RAM and Raspberry PI has 2GB Ram.

6.6 System Validation

As described earlier in Chapter 4, we have acquired data of arecanut from different views under five classes, viz Supari, Laal, Tukada, Kharad, and Baad, to build a dataset of 300 nuts (60 for each class). However, the segregation model described above is based on two classes, viz Good and Bad varieties. The justification for this two-class segregation is given in great detail in

section 6.1.1. Out of these 300 images, 60 belonging to Tukada were left out for the simple reason that they, being small pieces, may occupy a single slot of the segregation wheel and create confusion, leading to a long estimation time. Any way the Tukada variety gets sorted out during mechanical segregation buy drum wheel as described in Fig.6.3.

6.6.1 Re-Classification of Arecanuts

After the deduction of 60 images belonging to Tukada, from the remaining 240 images, Supari, Laal, and Kharad were put together in good variety, while the Baad class was put under Bad Variety. With these new classification rules, the ML algorithms loaded on Raspbery-5 re-trained for the identification of good and bad variety arecanuts. The selection rule was made quite simple here, in such a way that the unknown arecanut image acquisition and validation of the image is good or bad, can be done in the shortest time so that the speed of the arecanut segregation can be increased. For this, we adopted a selection rule, as any arecanut other than Bad is a Good variety. This machine-learning algorithm has to identify if the image presented before it is Bad, and if so, the hammer will knock the arecanut immediately to the bad basket. Rest all arecanuts will move to the good basket.

6.6.2 The Results of Classification

In all, 100 arecanuts were selected, of which 70 belonged to the good variety (Supari, Laal, and Kharad) and 30 belonged to the bad variety (Baad). All the arecanuts were mixed and poured in the feed funnel, as shown in Fig.6.4c. The segregation machine classified these arecanuts, and they were collected in respective trays (Good and Bad). To our expectation, all 30 bad variety arecanuts were collected in bad tray and the remaining 70 arecanuts in good tray. It took nearly 3 seconds to identify and segregate a given arecanut. Therefore, the speed of segregation is approximately 12,000 arecanut for a 10-hour operation. Considering an average weight of each arecanut equal to 15 grams, the segregation wheel can sort 180kg of arecanut in a 10-hour operation. The results were quite encouraging, and misclassification was down to zero. All the figures given here are based on the extrapolation of the 100-nuts classification and its overall speed of classification. Only care has to be taken that while feeding the arecanuts, we need to select the size of arecanut between 19mm-22mm so that they are accommodated comfortably in a given slot. The segregation speed can be increased many folds by using AI-enabled microcomputers in place of Raspberry Pi-5. One such platform has been recently launched by Intel subsidy. Altera's FPGA AI Suite and OpenVINO platforms use AI to generate

optimized intellectual property based on standard frameworks, such as TensorFlow and PyTorch, while its FPGAs are designed to integrate critical AI inferencing capabilities and better intercept evolving standards such as PCI Express, CXL, and 6G wirelesses. These new AI-based technologies can be integrated into the arecanut segregator to improve the sorting speed of the system at the same time, we can go for five-class segregation with some modifications in the system design.

6.6.3 Bill of Material:

Table 6.3 Bill of Materials

Sr. No	Item	Amount
1	Raspberry Pi-5	6,000/-
2	5MP Camrea	1,200/-
3	3Kg Stepper Motor..	1,500/-
4	Assorted Electronic components	600/-
5	Fabrication and other consumable	1,000/-
	Total	**10,300**

6.7 References:

1. Enhancing arecanut segregation with CNN-LSTM: A comparative analysis of DCNN and optimizers using Transfer learning" by Sameer Patil, Aparajita Naik, Marlon Sequeira and Jivan Parab' in Multimedia Tools and Applications (Springer) (SCOPUS) (Under review)

2.https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://medium.com/analytics-vidhya/a-complete-guide-to-adam-and-rmsprop-optimizer-75f4502d83be&ved=2ahUKEwiZ5eS7-oWJAxVr-TgGHaFpNuUQFnoECBwQAQ&usg=AOvVaw1X8vNb54OkWiNpCMG88vwB

3. "Machine vision Lights Boost Canning Line Inspection Speed" by Steve Kinney, in Vision Spectra, Laurin Publishing, Vol-6, Issue 2, 2024, pp10-12

4. Picture with courtesy from Goa Bagayatdar, Ponda, Goa

5. Srijan Motors, 25, Electronic Co - Op Estate, Pune - Satara Road, Pune - 411009 Maharashtra, India. Email Address : srijanmotors@gmail.com. Phone Number : +91-20-24225650

6. https://kitsguru.com/products/led-white-1w-high-brightness-and-high-power?variant=45213711925500&currency=INR&utm_medium=product_sync&utm_source=google&utm_content=sag_organic&utm_campaign=sag_organic&gad_source=1

7. Stepper Motors : Fundamentals, Applications And Design, V. V. Athani, New Age International, 1997 - Digital actuator - 201 pages

8. https://www.alldatasheet.com/datasheet-pdf/view/22400/STMICROELECTRONICS/IRF740.html

9. https://circuits-diy.com/wp-content/uploads/2021/08/UC3843.pdf

10. https://www.alldatasheet.com/datasheet-pdf/view/80066/INFINEON/SPP20N60S5.html

11. https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

12. https://www.ti.com/lit/ug/tidu737/tidu737.pdf

13. https://datasheets.raspberrypi.com/rpi5/raspberry-pi-5-product-brief.pdf

14. Intel launches stand-alone FPGA firm Altera, The News Wire in Vision Spectra Summer 2024 pp19-20

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

In this thesis, we try to address the various problems faced by small farmers of Goa, and the same solution may be adopted across the world, as arecanut is widely cultivated in Southeast Asia, South Africa, and the Pacific Ocean Islands. Arecanut has high commercial value and is used as a breath freshener, digestive aid, anthelmintic, aphrodisiac, and to maintain stamina. They have antioxidant characteristics, and their extracts are used to stop oxidative cell damage in normal cells. Studies reveal that anticancer compounds like flavonoid, polyphenol, carotenoid, selenium, vitamin C, and E, which showed chemotherapy and preventive activity, can be obtained from arecanuts. With all these medicinal properties, selecting a good quality arecanut from a bad one is a challenge. Here, we have successfully developed and deployed an arecanut segregation unit with almost 100 percent accuracy. The work is divided into mainly four parts. A) Develop a proper mechanism to collect the data of Good and Bad Arecanut. In this thesis, we have dealt with 5 classes of arecanuts, namely Supari, Laal, Kharad, Tukada, and Baad. B) Design and develop software to pre-process the data acquired so that good percentages of features are retained intact, and only these features are used for further processing. C) Compare various algorithms and develop custom ML algorithms to classify all these five categories of arecanut with a high percentage of accuracy and D) Design and Develop a prototype model of arecanut segregator which works on the above principle to demonstrate the feasibility of work undertaken for research work.

7.1 Development of Mechanism for Data Acquisition:

Here, we have designed and developed a custom manifold to acquire the images of the arecanuts with high quality. For such a work, we need proper white light illumination without any atmospheric disturbance. The acquired images must contain the true texture of the arecanuts, and full colour gamut must be represented. Fig 3.1 shows the custom setup, which is explained in depth in Chapter 3. As the final instrumentation was envisaged to be deployed on Raspberry Pi-5, the same platform was used for data acquisition as well. 300 images were acquired from five different varieties of arecanuts, which are normally bought by farmers to the traders for a cash sale. In this arrangement, the sample is kept at the bottom and a camera at the top. The sample is located about 14 cm below the camera using a hollow cylinder with interior sides covered using

black paper, as shown in Fig 4.2(a). All the acquisition of the images was done in a dark room to eliminate the variables due to environmental changes.

7.2 Pre-processing:

Image Pre-processing variety of operations were carried out on the images acquired in the previous steps. Improve the quality of the image depending on the application. These operations on the image aim to reduce the background noise and improve the image cross-section. For this purpose, a virtual mask was created only to cover the arecanut, and only the features inside the mask were sent for further processing. The colour of the background plays an important role, and hence, a complete matte dark background was created to enhance the arecanut contour. Three algorithms, namely, contrast enhancement, image filtering, and image segmentation. In contrast, enhancement algorithms, from Fig 4.4, we observed that CLAHE gives the best image contrast without deteriorating the image and maintaining good texture details. In Image Filtering algorithms, the anisotropic diffusion edge-preserving filter provides the best edge and texture preservation. Among the image segmentation algorithms, K-means segmentation was able to segment all images in the database efficiently. The overall analysis shows that when we combine the above three techniques, i.e., CLAHE, anisotropic diffusion edge-preserving filter, and K-means segmentation, we get the best possible results.

7.3 Classification Algorithms:

One of the main aims of the research work was to develop the most effective ML algorithm to classify the above described 5 classes of arecanuts. Here, we have developed a custom CNN, CNN-LSTM and used various pre-trained DCNN models like MobileNet, AlexNet, GoogLeNet, SqeezeNet, and NasNetMobile. The integration of CNNs and LSTMs in the CNN-LSTM architecture enables it to process both spatial and temporal information simultaneously. All these models were tried for various optimizers and different learning rates, and these were tuned for suitable hyperparameters. Fig 4.7 shows the system architecture of the proposed arecanut classification system. To evaluate the performance or quality of our model, we used metrics such as the Confusion matrix and AUC- ROC (Area under the Curve-Receiver Operating Characteristic). From the summary table presented in Chapter 4, it is observed that the CNN-LSTM model with RMSprop optimizer gave the highest accuracy of 99% with a precision of 99.10%. Similarly, the CNN-LSTM model with optimizer Adam and Sgdm gave an accuracy

of 98.33% and 98%, respectively. In all these studies, the number of Epochs was set to 50. In addition, the K-fold technique with 5 folds and 10 folds was used for model training. The CNN and MobileNet models achieved 100% F1-score. The models were also cross-validated using the 10-fold method and obtained an average 100% F1-score for MobileNet and 98% for CNN. MobileNet outperforms the CNN model as well as the literature reported work. We also tried to use AlexNet, however, its size of architecture was quite large compared to the customized CNN architecture. Hence, customized CNN will be more effective considering the required length of the conveyor belt and the required speed of segregation of the nuts when the automated system is designed. In addition to these algorithms, four pre-trained DCNN models, namely AlexNet, GoogleNet, SqeezeNet, and NasNetMobile, were also investigated for their efficacy. The results of these models are given in a comparative chart in Chapter 5. The limitation of the work is that the dataset has only healthy and diseased Arecanut images. In future work, more types of Arecanuts will be included in the dataset. The author will also try to implement other DL models, such as DenseNet201, VGG16, etc., to develop the system

7.4 Design and Development of a prototype:

One of the important take away from this research work is workable prototype model ready for field deployment after a proper packaging. The work highlights a single wheel-based arecanut segregator directly mounted on a 3-kg.cm stepper motor. The wheel at its outer periphery is fixed with 12 radial spokes of height 22mm. Thus, 12 compartments are formed in the outer diameter of the wheel. Each of these compartments (Slots) can comfortably accommodate 18mm to 22mm arecanut. A Pi camera mounted on the top of the slot takes the picture and sends it to the back-end microcomputer (Raspberry Pi-5). The design has been incorporated with many new features, such as an index hole for initializing the segregation wheel position after every single rotation to minimize the errors due to missed steps during the operation. The operation of the index hole in full detail is described in section 6.2.1. It has 24 volts solenoid activated hammer, as shown in Fig. 6.4c, to knock out bad arecanut from good ones. The design of the compartment size of the segregation wheel is a very important parameter in the entire design, as described in section 6.1.1. It may be noted here that the classification algorithms described in Chapter 4 have data acquisition and pre-processing software loaded in the Raspberry Pi-5. However, the complex ML algorithms are installed in a back-end highly resourceful computational architecture having CUDA (Compute Unified Distributed Architecture) support. The CUDA interacts with the NVIDIA graphics card (GEFORCE RTX

3060) to give the classification results in the shortest time. Here, we need to classify 5 different varieties of arecanut, and therefore, the classification software designed belongs to the multiclass segregation problem. The prototype segregation model is instead based on two-class problem (i.e. to identify good and bad arecanuts). Because of the simplified classification algorithm in this case, the whole classification software can be configured on the Raspberry Pi-5 microcomputer. The justification for moving from five-class arecanut segregation to two-class segregation is given in detail in section 6.1.1. The complete design described in Chapter 6 was fabricated and tested with different arecanuts other than those used for training and validation of arecanut images used in Chapter 4. In all, 100 arecanuts were selected, of which 70 belonged to the good variety (Supari, Laal, and Kharad) and 30 belonged to the bad variety (Baad). All the arecanuts were put together and poured in the funnel, as shown in Fig.6.4a. After the segregation, all of the arecanuts collected in tray identified for bad areca were compared with the original labels of these arecanuts and found to be belonging to the bad class of arecanuts. Similarly, all the arecanuts collected in tray identified as good contained only arecanuts having labeled as good before starting the operation of segregation. The 100 percent result showed that, the ML software designed for classification is most appropriate. On the issue of throughput of the system, we found that the machine with all its subsystems takes approximately 3 seconds on an average to identify and segregate an arecanut after it comes below the focus of the Pi Camera. With the extrapolation of the results, we can say the speed of segregation is approximately 12,000 arecanut for a 10-hour operation. On an average, an arecanut weighs 15 grams. Thus, we can confidently say the segregation wheel can sort 180kg of arecanut in a 10-hour operation. The results were quite encouraging and validated one of our important aims defined in Chapter 2.

7.5 OVERALL CONCLUSION

The research topic was chosen keeping in mind the hardship faced by arecanut farmers due to the delayed payments for their crops due to uncertainty and doubts in the minds of traders about the quality of the arecanuts. One cannot blame a trader for these delayed payments as it takes quite a long time to segregate good and bad arecanuts before they can make payment to the farmer. The primary aim of the work was to find a solution using state of art Artificial Inelegance technology (Identified by Govt. Of India as Disruptive Technology as per NEP 2020 actionable point 80). There were the main objectives namely a) Design of an image acquisition system for capturing quality images. b) Creating a database of different varieties of arecanuts c) Design of Deep Learning (DL) based algorithms. d) Development of AI-based

instrumentation for segregation, and e) Performance evaluation of the system. All these aims were investigated, and an appropriate solution was arrived at. We obtained a classification accuracy of 99% with a precision of 99.13% using the CNN-LSTM model with RMSprop optimizer. Having got an excellent result, we developed a prototype low-cost arecanut segregation model and was found to work efficiently even for untrained arecanut images. The two-class segregation developmental model gave almost 100% accurate results.

7.6 Future Scope:

The ML classification model developed in Chapter 4 was implemented on a backend desktop computer with high-end computational resources. In spite of such resources, the CNN-LSTM model took 27 minutes to give the results for single-fold iteration. Therefore, if one has made a portable system with 5 classifications, we need to switch our system to AI-enabled high-end processors such as the JETSON computing platform. Moreover, when you switch over to five-class segregation, we need to increase the number of hammers to four in a sequential pattern. This will require the change of segregation wheel design and the proportionate load competent stepper motors (or change over completely to servomotors).

In addition, we will need more hammers when we use 5-class segregation. Also, a finer detail of the arecanut is required to arrive at the correct category of arecanut. This means we will need all six-side views of the arecanut to properly classify a given arecanut. For example, the Laal variety of arecanut differs from the best quality arecanut (i.e. Supari) just by a tiny hole at the centre of its bottom. Therefore, the bottom view of the arecanut is of utmost importance. This arrangement will need multiple cameras placed at different angles to obtain the correct feature of an arecanut. This additional augmentation of cameras will lead to a more complex design of the arecanut segregation unit.

Further, one can add a load cell to determine the weight of the arecanut. The volume of the said arecanut can be determined by knowing the diameter of the arecanut from the contour of the arecanut, as described in Chapter 3. Thus, one can find the density of the arecanut and decide whether the given nut is good or bad. Normally, bad arecanuts are porous inside and weigh less comparatively.

In this research work, we have shown the possibility of arecanut segregation and with the availability of more sophisticated microcomputing platforms. Intel has launched Altera, a stand-alone field-programmable gate array (FPGA). Altera's FPGA AI Suite and OpenVINO platforms use AI to generate optimized intellectual property based on standard frameworks, such as

TensorFlow and PyTorch, while its FPGAs are designed to integrate critical AI inferencing capabilities and better intercept evolving standards. These new platforms can aid the development of faster and versatile arecanut segregator for seven classes. Though we have given in this thesis one set of solutions for the farmer's problems, better solutions can be worth pursuing; after all, our country’s economy is heavily dependent upon the farming community.

PUBLICATIONS

Journal Publications

1. “Efficient Deep Learning model for de-husked Arecanut classification” in Journal of Applied and Natural Science. JANS. 2023;15:1529–40. https://doi.org/10.31018/jans.v15i4.5067. (SCOPUS)
2. “Enhancing arecanut segregation with CNN-LSTM: A comparative analysis of DCNN and optimizers using Transfer learning” in Multimedia Tools and Applications (SCOPUS) (Under review).

Conference papers submitted/presented and published

1. “An Algorithm for Pre-processing of Arecanut for Quality Classification” at 02nd International Conference on Image Processing and Capsule Networks-ICIPCN-2021, vol. 300. Cham: Springer International Publishing; 2022. pp. 79–93. https://doi.org/10.1007/978-3-030-84760-9_8. (SCOPUS)
2. “A Dehusked Arecanut classification algorithm based on 10-fold cross-validation of Convolutional Neural Network” at the International Conference on Advanced Network Technologies and Intelligent Computing (ANTIC -2022) organized by BHU, India. Cham: Springer Nature Switzerland; 2023. pp. 35–45. https://doi.org/10.1007/978-3-031-28183-9_3. (SCOPUS)
3. "Optimizing Dehusked Arecanut Quality Segregation: CNN-Based Approach with Contrast Enhancement and Data Augmentation." At 3rd International Symposium on Smart Cities Challenges, Technologies, and Trends (SCCTT-2024) (Accepted)
4. “Comparison of throughput of ArecaNut segregation between CNN-LSTM and AlexNet” International Conference on Responsible Artificial Intelligence (ICRAI) 2024 to be held on 16 Dec.2024 at Mangalore University, Mangalore, Karnataka (communicated)(Scopus)

Other publications

1. “Enhancement of accuracy in soil urea estimation using machine learning tools” At 3rd International Symposium on Smart Cities Challenges, Technologies, and Trends (SCCTT-2024) (Accepted)

Appendix-1: AlexNet

```
root = "E:\dataset 23 APRIL 2021";
imds = imageDatastore(root, 'IncludeSubfolders', true, 'LabelSource', 'foldernames');
cv = cvpartition(imds.Labels, 'KFold', 5, 'Stratify', true);
% Load the pre-trained AlexNet model
net = alexnet;
;% Define class names
classNames = {'BAAD', 'KHARAD', 'LAAL', 'SUPARI', 'TUKADA'};
numClasses = numel(classNames);
r = 0;
for k = 1:5
    train_idx = training(cv, k);
    test_idx = test(cv, k);
    TRAIN = imageDatastore(imds.Files(train_idx));
    TRAIN.Labels = imds.Labels(train_idx);
    TRAIN.ReadFcn = @readFunctionTrain;
    TEST = imageDatastore(imds.Files(test_idx));
    TEST.Labels = imds.Labels(test_idx);
    TEST.ReadFcn = @readFunctionTrain;
    % Modify the Xception model to match the input size
    inputSize = [224 224 3];
    lgraph = layerGraph(net);
    lgraph = removeLayers(lgraph, {'loss3-classifier','prob', 'output'});
    layers = [
        fullyConnectedLayer(numClasses, 'Name', 'fc', ...
            'WeightLearnRateFactor', 20, 'BiasLearnRateFactor', 20)
        softmaxLayer('Name', 'softmax')
        classificationLayer('Name', 'classoutput', 'Classes', categorical(classNames)) %
Specify class names
    ];
    lgraph = addLayers(lgraph, layers);
    % Connect the layers
    lgraph = connectLayers(lgraph, 'pool5-drop_7x7_s1', 'fc');
    % Define training options
    opts = trainingOptions('adam', ...
        'InitialLearnRate', 0.0002, ...
        'Verbose', false, ...
        'MaxEpochs', 50, ...
        'MiniBatchSize', 16, ...
        'Shuffle', 'every-epoch', ...
        'Plots', 'training-progress', ...
        'ValidationData', TEST, ...
        'ValidationFrequency', 1);
    % Train the network
    [convNet, info] = trainNetwork(TRAIN, lgraph, opts);
       % Load test data
    [labels, score] = classify(convNet, TEST);
    confMat = confusionmat(TEST.Labels, labels);
    confMat = confMat ./ sum(confMat, 2);
    accuracy_na_nc(k) = mean(diag(confMat));
%similarity_scores = minmax_normalize_matrix(score)
if r==0
  save('cnn_no_contrast.mat','accuracy_na_nc','confMat');
  end
if r==1
  save('cnn_no_contrast1.mat','accuracy_na_nc','confMat');
end
if r==2
  save('cnn_no_contrast2.mat','accuracy_na_nc','confMat');
end
if r==3
  save('cnn_no_contrast3.mat','accuracy_na_nc','confMat');
  end
if r==4
  save('cnn_no_contrast4.mat','accuracy_na_nc','confMat');
```

```
end
if r==5
  save('cnn_no_contrast5.mat','accuracy_na_nc','confMat');
end
if r==6
  save('cnn_no_contrast6.mat','accuracy_na_nc','confMat');
end
if r==7
  save('cnn_no_contrast7.mat','accuracy_na_nc','confMat');
end
if r==8
  save('cnn_no_contrast8.mat','accuracy_na_nc','confMat');
end
if r==9
  save('cnn_no_contrast9.mat','accuracy_na_nc','confMat');
end
r=r+1;
    % Calculation recall, precision, and F1 score
    confMatt(:, :, k) = confMat'
    diagonal = diag(confMatt(:, :, k))
    sum_of_rows = sum(confMatt(:, :, k), 2)
    precision = diagonal ./ sum_of_rows
    overall_precision(k) = mean(precision)
    sum_of_columns = sum(confMatt(:, :, k), 1)
    recall = diagonal ./ sum_of_columns'
    overall_recall(k) = mean(recall)
    f1_score(k) = 2 * ((overall_precision(k) * overall_recall(k)) / (overall_precision(k) +
overall_recall(k)))
    % Plot confusion matrix
    figure
    confchrt = confusionchart(TEST.Labels, labels)
    str = sprintf('Classification of Areca Nut %d', k);
    confchrt.Title = str
    confchrt.NormalizedValues
    confchrt.RowSummary = 'row-normalized'
    confchrt.ColumnSummary = 'column-normalized'
    classNames = {'BAAD', 'KHARAD', 'LAAL', 'SUPARI', 'TUKADA'}
    figure
    % Plot ROC curve
    rocObj = rocmetrics(TEST.Labels, score, classNames)
    rocObj.AUC;
    plot(rocObj)
    hold on
    hold off;
end
function I=readFunctionTrain(filename)
I=imread(filename);
I=imresize(I,[227 227]);
end
```

Appendix-2: Adam_GoogleNet

```
root = "E:\dataset 23 APRIL 2021";
imds = imageDatastore(root, 'IncludeSubfolders', true, 'LabelSource', 'foldernames');
cv = cvpartition(imds.Labels, 'KFold', 5, 'Stratify', true);
% Load the pre-trained googlenet model
net = googlenet;
% Define class names
classNames = {'BAAD', 'KHARAD', 'LAAL', 'SUPARI', 'TUKADA'};
numClasses = numel(classNames);
r = 0;
for k = 1:5
    train_idx = training(cv, k);
    test_idx = test(cv, k);
    TRAIN = imageDatastore(imds.Files(train_idx));
    TRAIN.Labels = imds.Labels(train_idx);
    TRAIN.ReadFcn = @readFunctionTrain;
    TEST = imageDatastore(imds.Files(test_idx));
    TEST.Labels = imds.Labels(test_idx);
    TEST.ReadFcn = @readFunctionTrain;
    % Modify the Xception model to match the input size
    inputSize = [224 224 3];
    lgraph = layerGraph(net);
    lgraph = removeLayers(lgraph, {'loss3-classifier','prob', 'output'});
    layers = [
        fullyConnectedLayer(numClasses, 'Name', 'fc', ...
            'WeightLearnRateFactor', 20, 'BiasLearnRateFactor', 20)
        softmaxLayer('Name', 'softmax')
        classificationLayer('Name', 'classoutput', 'Classes', categorical(classNames)) %
Specify class names
    ];
    lgraph = addLayers(lgraph, layers);
    % Connect the layers
    lgraph = connectLayers(lgraph, 'pool5-drop_7x7_s1', 'fc');
    % Define training options
    opts = trainingOptions('adam', ...
        'InitialLearnRate', 0.0002, ...
        'Verbose', false, ...
        'MaxEpochs', 50, ...
        'MiniBatchSize', 16, ...
        'Shuffle', 'every-epoch', ...
        'Plots', 'training-progress', ...
        'ValidationData', TEST, ...
        'ValidationFrequency', 1);
    % Train the network
    [convNet, info] = trainNetwork(TRAIN, lgraph, opts);
       % Load test data
    [labels, score] = classify(convNet, TEST);
    confMat = confusionmat(TEST.Labels, labels);
    confMat = confMat ./ sum(confMat, 2);
    accuracy_na_nc(k) = mean(diag(confMat));
%similarity_scores = minmax_normalize_matrix(score)
if r==0
  save('cnn_no_contrast.mat','accuracy_na_nc','confMat');
  end
if r==1
  save('cnn_no_contrast1.mat','accuracy_na_nc','confMat');
  end
if r==2
  save('cnn_no_contrast2.mat','accuracy_na_nc','confMat');
  end
if r==3
  save('cnn_no_contrast3.mat','accuracy_na_nc','confMat');
  end
if r==4
  save('cnn_no_contrast4.mat','accuracy_na_nc','confMat');
```

```
end
if r==5
  save('cnn_no_contrast5.mat','accuracy_na_nc','confMat');
 end
if r==6
  save('cnn_no_contrast6.mat','accuracy_na_nc','confMat');
end
if r==7
  save('cnn_no_contrast7.mat','accuracy_na_nc','confMat');
end
if r==8
  save('cnn_no_contrast8.mat','accuracy_na_nc','confMat');
end
if r==9
  save('cnn_no_contrast9.mat','accuracy_na_nc','confMat');
end
r=r+1;
    % Calculation recall, precision, and F1 score
    confMatt(:, :, k) = confMat'
    diagonal = diag(confMatt(:, :, k))
    sum_of_rows = sum(confMatt(:, :, k), 2)
    precision = diagonal ./ sum_of_rows
    overall_precision(k) = mean(precision)
    sum_of_columns = sum(confMatt(:, :, k), 1)
    recall = diagonal ./ sum_of_columns'
    overall_recall(k) = mean(recall)
    f1_score(k) = 2 * ((overall_precision(k) * overall_recall(k)) / (overall_precision(k) +
overall_recall(k)))
    % Plot confusion matrix
    figure
    confchrt = confusionchart(TEST.Labels, labels)
    str = sprintf('Classification of Areca Nut %d', k);
    confchrt.Title = str
    confchrt.NormalizedValues
    confchrt.RowSummary = 'row-normalized'
    confchrt.ColumnSummary = 'column-normalized'
    classNames = {'BAAD', 'KHARAD', 'LAAL', 'SUPARI', 'TUKADA'}
    figure
    % Plot ROC curve
    rocObj = rocmetrics(TEST.Labels, score, classNames)
    rocObj.AUC;
    plot(rocObj)
    hold on
    hold off;
end
function I = readFunctionTrain(filename)
    I = imread(filename);
    I = imresize(I, [224, 224]);
end
% Custom classification layer to handle specific class names
function layer = customClassificationLayer(numClasses, classNames)
    layer = classificationLayer('Name', 'classoutput');
    layer.ClassNames = categorical(classNames);
end
```

Appendix-3: Adam_Mobilenetv2

```
root="Areca";
imds = imageDatastore(root, 'IncludeSubfolders', true, 'LabelSource', 'foldernames');
cv = cvpartition(imds.Labels, 'KFold', 5, 'Stratify', true);
 Load the pre-trained Xception model
net = mobilenetv2;
% Define class names
classNames = {'BAAD', 'KHARAD', 'LAAL', 'SUPARI', 'TUKADA'};
numClasses = numel(classNames);
r = 0;
for k = 1:5
    train_idx = training(cv, k);
    test_idx = test(cv, k);
    TRAIN = imageDatastore(imds.Files(train_idx));
    TRAIN.Labels = imds.Labels(train_idx);
    TRAIN.ReadFcn = @readFunctionTrain;
    TEST = imageDatastore(imds.Files(test_idx));
    TEST.Labels = imds.Labels(test_idx);
    TEST.ReadFcn = @readFunctionTrain;
    % Modify the Xception model to match the input size
    inputSize = [224 224 3];
    lgraph = layerGraph(net);
    lgraph = removeLayers(lgraph, {'Logits' ,  'Logits_softmax' ,
'ClassificationLayer_Logits'});
    layers = [
        fullyConnectedLayer(numClasses, 'Name', 'fc', ...
            'WeightLearnRateFactor', 20, 'BiasLearnRateFactor', 20)
        softmaxLayer('Name', 'softmax')
        classificationLayer('Name', 'classoutput', 'Classes', categorical(classNames)) %
Specify class names
    ];
    lgraph = addLayers(lgraph, layers);
    % Connect the layers
    lgraph = connectLayers(lgraph, 'global_average_pooling2d_1', 'fc');
    % Define training options
    opts = trainingOptions('adam', ...
        'InitialLearnRate', 0.0002, ...
        'Verbose', false, ...
        'MaxEpochs', 50, ...
        'MiniBatchSize', 16, ...
        'Shuffle', 'every-epoch', ...
        'Plots', 'training-progress', ...
        'ValidationData', TEST, ...
        'ValidationFrequency', 1);
    % Train the network
    [convNet, info] = trainNetwork(TRAIN, lgraph, opts);
       % Load test data
    [labels, score] = classify(convNet, TEST);
    confMat = confusionmat(TEST.Labels, labels);
    confMat = confMat ./ sum(confMat, 2);
    accuracy_na_nc(k) = mean(diag(confMat));
    % Save the results
   if r==0
  save('cnn_no_contrast.mat','accuracy_na_nc','confMat');

end
if r==1
  save('cnn_no_contrast1.mat','accuracy_na_nc','confMat');
  end
if r==2
  save('cnn_no_contrast2.mat','accuracy_na_nc','confMat');
  end
if r==3
  save('cnn_no_contrast3.mat','accuracy_na_nc','confMat');
  end
```

```
if r==4
  save('cnn_no_contrast4.mat','accuracy_na_nc','confMat');
end
if r==5
  save('cnn_no_contrast5.mat','accuracy_na_nc','confMat');
 end
if r==6
  save('cnn_no_contrast6.mat','accuracy_na_nc','confMat');
end
if r==7
  save('cnn_no_contrast7.mat','accuracy_na_nc','confMat');
end
if r==8
  save('cnn_no_contrast8.mat','accuracy_na_nc','confMat');
end
if r==9
  save('cnn_no_contrast9.mat','accuracy_na_nc','confMat');
end
r=r+1;
    % Calculation recall, precision, and F1 score
    confMatt(:, :, k) = confMat'
    diagonal = diag(confMatt(:, :, k))
    sum_of_rows = sum(confMatt(:, :, k), 2)
    precision = diagonal ./ sum_of_rows
    overall_precision(k) = mean(precision)
    sum_of_columns = sum(confMatt(:, :, k), 1)
    recall = diagonal ./ sum_of_columns'
    overall_recall(k) = mean(recall)
    f1_score(k) = 2 * ((overall_precision(k) * overall_recall(k)) / (overall_precision(k) +
overall_recall(k)))
    % Plot confusion matrix
    figure
    confchrt = confusionchart(TEST.Labels, labels)
    str = sprintf('Classification of Areca Nut %d', k);
    confchrt.Title = str
    confchrt.NormalizedValues
    confchrt.RowSummary = 'row-normalized'
    confchrt.ColumnSummary = 'column-normalized'
    classNames = {'BAAD', 'KHARAD', 'LAAL', 'SUPARI', 'TUKADA'}
    figure
    % Plot ROC curve
    rocObj = rocmetrics(TEST.Labels, score, classNames)
    rocObj.AUC;
    plot(rocObj)
    hold on
    hold off
end
function I = readFunctionTrain(filename)
    I = imread(filename);
    I = imresize(I, [224, 224]);
end
```

Appendix-4: Adam_NasNetmobile

```
root="dataset 23 APRIL 2021";
imds = imageDatastore(root, 'IncludeSubfolders', true, 'LabelSource', 'foldernames');
cv = cvpartition(imds.Labels, 'KFold', 5, 'Stratify', true);
% Load the pre-trained Xception model
net = nasnetmobile;
% Define class names
classNames = {'BAAD', 'KHARAD', 'LAAL', 'SUPARI', 'TUKADA'};
numClasses = numel(classNames);
r = 0;
for k = 1:5
    train_idx = training(cv, k);
    test_idx = test(cv, k);
    TRAIN = imageDatastore(imds.Files(train_idx));
    TRAIN.Labels = imds.Labels(train_idx);
    TRAIN.ReadFcn = @readFunctionTrain;
    TEST = imageDatastore(imds.Files(test_idx));
    TEST.Labels = imds.Labels(test_idx);
    TEST.ReadFcn = @readFunctionTrain;
    % Modify the Xception model to match the input size
    inputSize = [224 224 3];
    lgraph = layerGraph(net);
    lgraph = removeLayers(lgraph, {'predictions' ,  'predictions_softmax' ,
'ClassificationLayer_predictions'});
    layers = [
        fullyConnectedLayer(numClasses, 'Name', 'fc', ...
            'WeightLearnRateFactor', 20, 'BiasLearnRateFactor', 20)
        softmaxLayer('Name', 'softmax')
        classificationLayer('Name', 'classoutput', 'Classes', categorical(classNames)) %
Specify class names
    ];
    lgraph = addLayers(lgraph, layers);
    % Connect the layers
    lgraph = connectLayers(lgraph, 'global_average_pooling2d_1', 'fc');
    % Define training options
    opts = trainingOptions('adam', ...
        'InitialLearnRate', 0.0002, ...
        'Verbose', false, ...
        'MaxEpochs', 50, ...
        'MiniBatchSize', 16, ...
        'Shuffle', 'every-epoch', ...
        'Plots', 'training-progress', ...
        'ValidationData', TEST, ...
        'ValidationFrequency', 1);
    % Train the network
    [convNet, info] = trainNetwork(TRAIN, lgraph, opts);
       % Load test data
    [labels, score] = classify(convNet, TEST);
    confMat = confusionmat(TEST.Labels, labels);
    confMat = confMat ./ sum(confMat, 2);
    accuracy_na_nc(k) = mean(diag(confMat));
    % Save the results
   if r==0
  save('cnn_no_contrast.mat','accuracy_na_nc','confMat');

end
if r==1
  save('cnn_no_contrast1.mat','accuracy_na_nc','confMat');

end
if r==2
  save('cnn_no_contrast2.mat','accuracy_na_nc','confMat');

end
if r==3
```

```
  save('cnn_no_contrast3.mat','accuracy_na_nc','confMat');

end
if r==4
  save('cnn_no_contrast4.mat','accuracy_na_nc','confMat');
end
if r==5
  save('cnn_no_contrast5.mat','accuracy_na_nc','confMat');
 end
if r==6
  save('cnn_no_contrast6.mat','accuracy_na_nc','confMat');
end
if r==7
  save('cnn_no_contrast7.mat','accuracy_na_nc','confMat');
end
if r==8
  save('cnn_no_contrast8.mat','accuracy_na_nc','confMat');
end
if r==9
  save('cnn_no_contrast9.mat','accuracy_na_nc','confMat');
end
r=r+1;
    % Calculation recall, precision, and F1 score
    confMatt(:, :, k) = confMat'
    diagonal = diag(confMatt(:, :, k))
    sum_of_rows = sum(confMatt(:, :, k), 2)
    precision = diagonal ./ sum_of_rows
    overall_precision(k) = mean(precision)
    sum_of_columns = sum(confMatt(:, :, k), 1)
    recall = diagonal ./ sum_of_columns'
    overall_recall(k) = mean(recall)
    f1_score(k) = 2 * ((overall_precision(k) * overall_recall(k)) / (overall_precision(k) +
overall_recall(k)))
    % Plot confusion matrix
    figure
    confchrt = confusionchart(TEST.Labels, labels)
    str = sprintf('Classification of Areca Nut %d', k);
    confchrt.Title = str
    confchrt.NormalizedValues
    confchrt.RowSummary = 'row-normalized'
    confchrt.ColumnSummary = 'column-normalized'
    classNames = {'BAAD', 'KHARAD', 'LAAL', 'SUPARI', 'TUKADA'}
    figure
    % Plot ROC curve
    rocObj = rocmetrics(TEST.Labels, score, classNames)
    rocObj.AUC;
    plot(rocObj)
    hold on
    hold off
end
function I = readFunctionTrain(filename)
  I = imread(filename);
  I = imresize(I, [227, 227]);
end
% Custom classification layer to handle specific class names
function layer = customClassificationLayer(numClasses, classNames)
  layer = classificationLayer('Name', 'classoutput');
  layer.ClassNames = categorical(classNames);
end
```

Appendix-5: Adam_SqueezeNet

```
root = "E:\dataset 23 APRIL 2021";
imds = imageDatastore(root, 'IncludeSubfolders', true, 'LabelSource', 'foldernames');
cv = cvpartition(imds.Labels, 'KFold', 5, 'Stratify', true);
% Load the pre-trained Xception model
net = squeezenet;
% Define class names
classNames = {'BAAD', 'KHARAD', 'LAAL', 'SUPARI', 'TUKADA'};
numClasses = numel(classNames);
r = 0;
for k = 1:5
    train_idx = training(cv, k);
    test_idx = test(cv, k);
    TRAIN = imageDatastore(imds.Files(train_idx));
    TRAIN.Labels = imds.Labels(train_idx);
    TRAIN.ReadFcn = @readFunctionTrain;
    TEST = imageDatastore(imds.Files(test_idx));
    TEST.Labels = imds.Labels(test_idx);
    TEST.ReadFcn = @readFunctionTrain;
    % Modify the Xception model to match the input size
    inputSize = [227 227 3];
    lgraph = layerGraph(net);
    lgraph = removeLayers(lgraph, {'prob', 'ClassificationLayer_predictions'});
    layers = [
        fullyConnectedLayer(numClasses, 'Name', 'fc', ...
            'WeightLearnRateFactor', 20, 'BiasLearnRateFactor', 20)
        softmaxLayer('Name', 'softmax')
        classificationLayer('Name', 'classoutput', 'Classes', categorical(classNames)) %
Specify class names
    ];
    lgraph = addLayers(lgraph, layers);
    % Connect the layers
    lgraph = connectLayers(lgraph, 'pool10', 'fc');
    % Define training options
    opts = trainingOptions('adam', ...
        'InitialLearnRate', 0.0002, ...
        'Verbose', false, ...
        'MaxEpochs', 50, ...
        'MiniBatchSize', 16, ...
        'Shuffle', 'every-epoch', ...
        'Plots', 'training-progress', ...
        'ValidationData', TEST, ...
        'ValidationFrequency', 1);
    % Train the network
    [convNet, info] = trainNetwork(TRAIN, lgraph, opts);
       % Load test data
    [labels, score] = classify(convNet, TEST);
    confMat = confusionmat(TEST.Labels, labels);
    confMat = confMat ./ sum(confMat, 2);
    accuracy_na_nc(k) = mean(diag(confMat));
    % Save the results
    if r==0
  save('cnn_no_contrast.mat','accuracy_na_nc','confMat');
  end
if r==1
  save('cnn_no_contrast1.mat','accuracy_na_nc','confMat');
  end
if r==2
  save('cnn_no_contrast2.mat','accuracy_na_nc','confMat');
  end
if r==3
  save('cnn_no_contrast3.mat','accuracy_na_nc','confMat');
  end
if r==4
  save('cnn_no_contrast4.mat','accuracy_na_nc','confMat');
```

```
end
if r==5
  save('cnn_no_contrast5.mat','accuracy_na_nc','confMat');
 end
if r==6
  save('cnn_no_contrast6.mat','accuracy_na_nc','confMat');
end
if r==7
  save('cnn_no_contrast7.mat','accuracy_na_nc','confMat');
end
if r==8
  save('cnn_no_contrast8.mat','accuracy_na_nc','confMat');
end
if r==9
  save('cnn_no_contrast9.mat','accuracy_na_nc','confMat');
end
r=r+1;
    % Calculation recall, precision, and F1 score
    confMatt(:, :, k) = confMat'
    diagonal = diag(confMatt(:, :, k))
    sum_of_rows = sum(confMatt(:, :, k), 2)
    precision = diagonal ./ sum_of_rows
    overall_precision(k) = mean(precision)
    sum_of_columns = sum(confMatt(:, :, k), 1)
    recall = diagonal ./ sum_of_columns'
    overall_recall(k) = mean(recall)
    f1_score(k) = 2 * ((overall_precision(k) * overall_recall(k)) / (overall_precision(k) +
overall_recall(k)))
    % Plot confusion matrix
    figure
    confchrt = confusionchart(TEST.Labels, labels)
    str = sprintf('Classification of Areca Nut %d', k);
    confchrt.Title = str
    confchrt.NormalizedValues
    confchrt.RowSummary = 'row-normalized'
    confchrt.ColumnSummary = 'column-normalized'
    classNames = {'BAAD', 'KHARAD', 'LAAL', 'SUPARI', 'TUKADA'}
    figure
    % Plot ROC curve
    rocObj = rocmetrics(TEST.Labels, score, classNames)
    rocObj.AUC;
    plot(rocObj)
    hold on
    hold off;
endfunction I = readFunctionTrain(filename)
    I = imread(filename);
    I = imresize(I, [227, 227]);
end
% Custom classification layer to handle specific class names
function layer = customClassificationLayer(numClasses, classNames)
    layer = classificationLayer('Name', 'classoutput');
    layer.ClassNames = categorical(classNames);
end
```

Appendix-6: CNN +LSTM_adam

```
root="dataset 23 APRIL 2021";
imds = imageDatastore(root,'IncludeSubfolders',true, 'LabelSource', 'foldernames');
cv = cvpartition(imds.Labels, 'KFold', 5, 'Stratify', true); r=0;
for k=1:5
    train_idx=training(cv,k); test_idx=test(cv,k);
    TRAIN=imageDatastore(imds.Files(train_idx));
    TRAIN.Labels=imds.Labels(train_idx);
    TRAIN.ReadFcn=@readFunctionTrain ;
    TEST=imageDatastore(imds.Files(test_idx));
    TEST.Labels=imds.Labels(test_idx);
    TEST.ReadFcn=@readFunctionTrain ;
%DEFINE LAYERS
  layers = [
        imageInputLayer([227 227 3])
        convolution2dLayer(3,8,'Padding','same')
        batchNormalizationLayer
        reluLayer
        averagePooling2dLayer(2,'Stride',2)
                convolution2dLayer(3,16,'Padding','same')
        batchNormalizationLayer
        reluLayer
        averagePooling2dLayer(2,'Stride',2)
                convolution2dLayer(3,32,'Padding','same')
        batchNormalizationLayer
        reluLayer
        averagePooling2dLayer(2,'Stride',2)
                convolution2dLayer(3,64,'Padding','same')
        batchNormalizationLayer
        reluLayer
        averagePooling2dLayer(2,'Stride',2)
               convolution2dLayer(3,128,'Padding','same')
        batchNormalizationLayer
        reluLayer
        averagePooling2dLayer(2,'Stride',2)
        convolution2dLayer(3,128,'Padding','same')
        batchNormalizationLayer
        reluLayer
        averagePooling2dLayer(2,'Stride',2)
        convolution2dLayer(3,256,'Padding','same')
        batchNormalizationLayer
        reluLayer
        dropoutLayer(0.2)
        flattenLayer('Name','flatten')
        lstmLayer(100, 'OutputMode', 'last')
        fullyConnectedLayer(5)
        softmaxLayer
        classificationLayer];
        %DEFINE TRAINING OPTIONS
opts = trainingOptions('adam', ...
    'InitialLearnRate', 0.0002, ...
    'Verbose',false, ...
    'MaxEpochs', 50, ...  %epochs has been changed
    'MiniBatchSize',16, 'Shuffle', 'every-epoch', 'Plots', 'training- ...
    progress',ValidationData=TEST,ValidationFrequency=1);
 %TRAIN NETWORK
[convNet, info] = trainNetwork(TRAIN, layers, opts);
%LOAD TEST DATA
[labels, score] = classify(convNet, TEST);
confMat = confusionmat(TEST.Labels, labels);
```

```
confMat = confMat./sum(confMat,2);
accuracy_na_nc(k) = mean(diag(confMat))
% similarity_scores = minmax_normalize_matrix(score)
if r==0
  save('cnn_no_contrast.mat','accuracy_na_nc','confMat');
  end
if r==1
  save('cnn_no_contrast1.mat','accuracy_na_nc','confMat');
  end
if r==2
  save('cnn_no_contrast2.mat','accuracy_na_nc','confMat');
  end
if r==3
  save('cnn_no_contrast3.mat','accuracy_na_nc','confMat');
  end
if r==4
  save('cnn_no_contrast4.mat','accuracy_na_nc','confMat');
end
if r==5
  save('cnn_no_contrast5.mat','accuracy_na_nc','confMat');
 end
if r==6
  save('cnn_no_contrast6.mat','accuracy_na_nc','confMat');
end
if r==7
  save('cnn_no_contrast7.mat','accuracy_na_nc','confMat');
end
if r==8
  save('cnn_no_contrast8.mat','accuracy_na_nc','confMat');
end
if r==9
  save('cnn_no_contrast9.mat','accuracy_na_nc','confMat');
end
r=r+1;
% Calculation recall, precision and F1 score
confMatt(:,:,k)= confMat'
diagonal = diag(confMatt(:,:,k))
sum_of_rows = sum(confMatt(:,:,k), 2)
precision = diagonal ./ sum_of_rows
overall_precision(k) = mean(precision)
sum_of_columns = sum(confMatt(:,:,k), 1)
recall = diagonal ./ sum_of_columns'
overall_recall(k) = mean(recall)
f1_score(k) =
2*((overall_precision(k)*overall_recall(k))/(overall_precision(k)+overall_recall(k)))
figure
confchrt = confusionchart(TEST.Labels, labels)
str =sprintf('Classification of Areca Nut %d',k);
confchrt.Title = str
confchrt.NormalizedValues
confchrt.RowSummary = 'row-normalized'
confchrt.ColumnSummary = 'column-normalized'
classNames = {'BAAD', 'KHARAD', 'LAAL', 'SUPARI', 'TUKADA'}; figure
rocObj = rocmetrics(TEST.Labels, score,classNames);
rocObj.AUC;plot(rocObj);hold on; hold off
end
function I=readFunctionTrain(filename)
I=imread(filename);
I=imresize(I,[227 227]);
end
```