T.R.

GEBZE TECHNICAL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

DEFECTIVE FRUITS DETECTION USING DEEP CONVOLUTIONAL NETWORKS

RHUDIE GRACE ZANG EDZANG

A THESIS SUBMITTED FOR THE DEGREE OF

MASTER OF SCIENCE

DEPARTMENT OF COMPUTER ENGINEERING

GEBZE

2019

T.R.
GEBZE TECHNICAL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

DEFECTIVE FRUITS DETECTION USING DEEP CONVOLUTIONAL NETWORKS

RHUDIE GRACE ZANG EDZANG

A THESIS SUBMITTED FOR THE DEGREE OF
MASTER OF SCIENCE
DEPARTMENT OF COMPUTER ENGINEERING

THESIS SUPERVISOR
ASSOC.PROF. DR. MEHMET GÖKTÜRK

GEBZE
2019

T.C.

GEBZE TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

DERİN KONVOLÜSYON AĞLARI İLE BOZULMUŞ MEYVELERİN TESPİTİ

RHUDIE GRACE ZANG EDZANG

YÜKSEK LİSANS TEZİ

BILGISAYAR MÜHENDİSLİĞİ ANABİLİM DALI

DANIŞMANI

DOÇ. DR. MEHMET GÖKTÜRK

GEBZE

2019

GEBZE TEKNİK ÜNİVERSİTESİ	**YÜKSEK LİSANS JÜRİ ONAY FORMU**

GTÜ Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 10/07/2019 tarih ve 2019/31 sayılı kararıyla oluşturulan jüri tarafından 17..../.10..../.2019. tarihinde tez savunma sınavı yapılan Rhudie Grace ZANG EDZANG'ın tez çalışması BILGISAYAR Mühendisliği Anabilim Dalında YÜKSEK LİSANS tezi olarak kabul edilmiştir.

JÜRİ

ÜYE
(TEZ DANIŞMANI) : Doç. Dr. Mehmet GÖKTÜRK

ÜYE : Doç. Dr. Erchan APTOULA

ÜYE : Doç. Dr.Onur DEMIR

ONAY

Gebze Teknik Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun

....../....../......... tarih ve ..../........ sayılı kararı.

ABSTRACT

Image classification and object detection are indispensable tasks of computer vision. Over the past few years, these tasks have been largely accomplished by deep neural networks (CNNs) while the interest in this powerful technique has increased in many areas as well. Most datasets consist of RGB images and these images are used as input images without modifications. In this thesis, the main focus is on the classification of fruit disease images. Although fruit disease recognition has become an increasingly popular topic of research, it still remains a scientific and technical challenge. Exploration and study of the importance of color spaces on fruit classification performance were as well looked in to. In our model, we principally use a concatenation of three-color space as an input image, on the other hand, we made an analysis with different single color spaces to see what kind of influence the color spaces have on the performance of fruit classification in terms of accuracy. The transformations are mainly summarized by the conversion of RGB images into HSV and CIELAB color spaces. In conclusion, changing the RGB into different color spaces can have a global impact on the training process. Using multiple color spaces as input of a CNN can significantly affect the accuracy of fruit classification by giving excellent results, sometimes better than the single color space. Different approaches have been studied in this thesis as well and the fusion approach performance works quite well in comparison with other approaches. Our approach been the fusion approach achieved the best, with 91.23% for the Custom-CNN and 97.94% for the Alex-Net, while 84.20% was obtained for the transfer learning and 71.60% for the classical image processing approach. However, using a model that allows multiple color spaces requires significant amount of time and resources because of the preprocessing step (converting to each color space) but, preprocessing can be done in real-time if adequate resources are provided.

Keywords: Classification, Deep features Extraction, Fruits diseases, Image processing, color spaces.

ÖZET

Görüntülerin sınıflandırılması ve tanınması, bilgisayarlı görmenin vazgeçilmez bir görevidir. Son yıllarda, bu görevler büyük ölçüde, derin sinir ağları (CNN'ler) tarafından yerine getirilmiş, bu güçlü tekniğe ilgi birçok alanda artmıştır.Çoğu veri kümesi, RGB görüntülerden oluşur ve bu görüntüler modifikasyon olmadan giriş görüntüleri olarak kullanılmaktadır. Bu tezde, esas olarak meyve hastalığı görüntülerinin sınıflandırılması ve tanınmasına odaklanılmıştır. Meyve tanıma giderek daha popüler bir araştırma konusu haline gelmesine rağmen, bilimsel ve teknik bir zorluk olmaya devam etmektedir. Meyve sınıflandırmalarının renk uzaylarının önemi de de tartışıldı ve incelendi.Modelimizde prensip olarak, bir giriş görüntüsü olarak üç renk uzayının birleşimi kullanıldı; diğer yandan, renk uzaylarının meyve sınıflandırma performansının doğruluğunu nasıl bir etkisi olabileceğini görmek için farklı tek renk uzaylarıyla analizler yapıldı. Dönüşümler temel olarak RGB görüntülerini HSV ve CIELAB renk uzaylarına dönüştürerek özetlenir.Sonuç olarak, RGB yi farklı renk alanlarına dönüştürmek öğrenme süreci üzerinde genel bir etkiye sahip olabilir.Bir CNN'nin girişi olarak çoklu renk uzayları kullanmak, bazen tek renkten daha iyi mükemmel sonuçlar vererek renk alanı sınıflandırma doğruluğunu önemli ölçüde etkileyebilmektedir. Bu tezde, çalışmasında farklı yaklaşımlar da çalışılmış ve füzyon yaklaşımının performansı diğer yaklaşımlarla karşılaştırıldığında iyi çalışmıştır.

Birleşme yaklaşımımız iyi çalıştı, Custom-CNN için %91.23 ve Alex-Net için %97.94, transfer öğrenimi için %84.20 ve klasik görüntü işleme yaklaşımı için %71.60.Bununla birlikte, çoklu renk aralığına izin veren bir model kullanmak, ön işleme aşamasının (her renk alanına dönüştürme) kendi başına büyük miktarda kaynak harcadığı için önemli miktarda zaman gerektirmektedir. Ancak, yeterli kaynak sağlanmışsa, ön işleme gerçek zamanlı olarak gerçekleştirilebilir.

Anahtar Kelimeler: Derin Öğrenme, Daha Hızlı RCNN, Makine Öğrenimi, Satış El ilanları, Nesne Tespiti, Metin Tespiti.

ACKNOWLEDGEMENTS

The accomplishment of this modest work is the result of a number of events, meetings, discussions and very fruitful relationships that allowed me to surround myself with human resources as well as to acquire the necessary scientific knowledge for the realization of this thesis. For this reason, it is very difficult to count the number of people to whom I owe thanks to!. First, I thank God Almighty, for giving me the strength to survive, as well as the audacity to overcome all difficulties. I would love to say a few words of thanks to my supervisor Doç. Dr. Mehmet GÖKTÜRK, who managed to guide, supervise me with efficiency, insight and a lot of patience during this period. I express my deep gratitude for his support and encouragement but also for his availability. These thanks do not only concern your scientific skills because talent is nothing without vision and good management. I would also like to say a big thanks to Doç. Dr. Erchan APTOULA for his availability, his support, his valuable advices, and his patience. A great thanks to the administrative staff who have always been helpful, kind, smiling and most importantly professional, for their role and implication in my academic evolution. I especially thank my friends, with whom I shared some very pleasant and sometimes difficult moments with. For me, this chapter of my life will engraved forever in my thoughts because, despite our ethnic, cultural and other differences, we were able to share so many things, thank you for allowing me to think of something else than my thesis during these three years."Mohammed fouceni, Muhtar, Zubeda, Wuday, Harliton, Cima, Odongo, Nour". I thank Philip for his time, his patience, his help, his critics, thank you for existing. I desire to express my deepest thanks to my dear parents. Dad, mom, it's very difficult to thank you because, it goes beyond the scope of this thesis, I thank you for everything you did for me in life. Finally, my deepest thanks go to Axel Gedeon MENGARA who is my source of inspiration and trusty. Thank you for making me what I am today and I hope you will have a long life so that we can share other successes together. I obviously have a very particular thought for my family, I purely and simply say: thank you for being in the garden of my life. Finally, I thank everyone who contributed directly or indirectly to this work.

TABLE of CONTENTS

LIST of ABBREVIATIONS and ACRONYMS

Abbreviations and Acronyms		**Explanations**
ANN	:	Artificial Neural Network.
CNN	:	Convolutional Neural Network.
CPU	:	Central Processing Unit.
DIP	:	Digital Image Processing.
FC	:	Fully Connected.
GCH	:	Global Color Histogram.
GPU	:	Graphics Processing Unit.
HD	:	High Definition.
KNN	:	K-Nearest Neighbors.
LBP	:	Local Binary Pattern.
ML	:	Machine Learning.
ReLU	:	Rectified-Linear unit.
ROI	:	Region of Interest.
SVM	:	Support Vector Machine.

LIST of FIGURES

LIST of TABLES

1. INTRODUCTION AND MOTIVATIONS

This section provides an overview of current trends in technology and applications related to fruit and vegetable disease detection. Allowing us to situate our thesis topic in its general context.

- Introduction

Fruits are a very important part of a healthy diet, as they are a proven source of vitamins and other nutrients that are beneficial to humans and therfore access to fresh and disease free fruits and vegetables became an important concern that cannot be ignored. The interest of the health science community in detecting diseases in agricultural products have increased significantly over the years. Researchers, decision-makers, and students share a common goal that for correctly identify diseases as soon as they appear, but this interest is also to prevent:

- Damage of the harvest.
- Reduce the quantity and quality of production.
- Increase production costs.

And all this can be done thanks to:

- Advanced HD camera.
- High-performance processors.
- Image processing and learning techniques.

- General Context

The goal of an inspection system is, first and foremost, to detect failures quickly and automatically in order to avoid unsafe operating conditions, increase availability and productivity and to ensure product quality. Although, very active in the field of research, the recognition and classification of fruits and vegetables remain a scientific subject with multiple challenges. The title of this thesis is "Defective Fruits Detection Using Deep Convolutional Networks" and our work focuses on image classification of agricultural products. It aims to design an architecture for the detection and classification of fruit and vegetable diseases. In agriculture, the visual quality control of products is unavoidable but with image processing, the image serves as a tool that enables technology to replace and even surpass the traditional means (human eyes) of fruit control. Computer vision on the other hand, is the technology that allows us to replace human vision in the process of fruit inspection and grading, it also allows us to do it automatically. In the process of fruits inspection, an automatic inspection system can be described by the combination of sophisticated image processing and advanced computer vision techniques. This coupling has a great reputation of being the means for excellence to offer good results and rapid identification thus, reducing man's efforts and the cost of labor. To find an efficient and effective solution to this problem, many approaches have been proposed for identifying and classifying fruits by their diseases (e.g. apples, dates, oranges, and other citrus fruits). Then, many surveys and comparative studies on the classification and identification of fruit diseases based on image analysis techniques can be found in the literature[1]-[10]. As we said at the beggining of this introduction our work is to focus on classification of fruit disease images, In addition, we investigated and evaluated the effect of color space transformation on fruits classification performance and finding the appropriate color space for fruit classification.

- Research Objectives

In a general framework, the purpose of this work is to try to measure the impact caused by diseases in fruits, more specifically, the study seeks to:

i) Evaluate the dependency of color in the detection and classification of fruit diseases.
ii) Compare the different methods of detection and classification of fruits.

- Structure of the Study

Before going further, the outline of the thesis is given as follows.
Chapter 2 presents the state of the art and has 6 sections.

- Section 2.1: Gives an overview of what has been done in the literature and state of art.
- Section 2.2: Fruits and Vegetables Diseases Detection and Recognition.
- Section 2.3: Deep learning and Agriculture.

- Section 2.4: History of Deep Learning.
- Section 2.5: The Convolutional Neural Network in Place of the Classical Approach.
- Section 2.6: Transfer Learning.

Chapter 3 corresponds to our theoretical contributions and has 5 sections.

- Section3.1: Gives a background about Neural Network.
- Section 3.2 : Gives a background about Convolutional Neural Network.
- Section 3.3: Presents the different architectures of CNN which have been used in this thesis.
- Section 3.4: Gives a background on color spaces.
- Section 3.5: Presents the architecture of the proposed model.

Chapter 4 concerns the implementation of the different methods that have been used.

- Section 4.1: Gives a background about the software,libraries and materials used.
- Section 4.2: Presents the dataset,and all the process information on data processing.
- Section 4.3: Presents the implementation of the different models that have been chosen for this thesis.
- Section 4.4: Presents the implementation of the diffrerent classical approaches.

Chapter 5 is about the presentation and evaluation of the results obtained.

Chapter 6 concludes the thesis on a global discussion of methodologies, results and discusses the work in perspective.

2. STATE OF THE ART

This second chapter is intended to cover the work of the existing literature for the recognition and detection of fruit diseases. The subject discussed in this section, has been developed by many researches in recent years so, this chapter gives us the opportunity to review the different approaches and methods established to overcome the problem of fruit diseases classification.

2.1. Fruits Detection and Classification

Detection and classification are learning-related concepts that define the acquisition of knowledge and the reuse of this new knowledge. For us humans, our learning is done throughout our lives, we learn from the perception of the environment with the five senses, experiences of life, the repetition of events with memory and our judgment (free will or intelligence) which is completely different for machines, because they do not have meaning or dynamic judgment, they obey the instructions of a program with input data and a program output or response.

Using many algorithms, the computer is able to recognize the pattern of an image. In the agricultural sector, the standard method for obtaining these results and satisfactory results is to use unsupervised segmentation, followed by feature extraction and finally classification using some algorithms of machine learning like support vector machine [5].

2.2. Fruits and Vegetables Diseases Detection and Recognition

The process used for the detection of diseases is similar to that explained above. The difference is that a step is added, preprocessing part [12], it embellishes the quality of the image and makes the infected part more visible then, the next step which is the segmentation step, concerns the region of interest of detection (ROI) which is primarily the part that contains the disease thanks to many segmentation techniques. Next, come the feature extraction step with some computer vision techniques including Local and complete local binary pattern [8], color histogram, deep features [9]. and classification with machine learning models like support vector machine (SVM)[1], multi-class SVM (M-SVM) [9], artificial neural network (ANN)[13][10], and Bayesian network [11]. Some variations can change the classification algorithm, seen in [14], [15] where the k-Nearest neighbor is used. We have been able to summarize the different techniques used in different articles. which can be observed in Table 2.1. Regarding the accuracy of the different techniques used for the recognition of fruit diseases, the formula is given below:

$$\text{Accuray} = \frac{\text{total number of the image correctly identify}}{\text{total number of the image used for testing}} * 100 \tag{2.1}$$

The traditional method for the recognition of fruits can be summarized as Pre-processing_Segmentation_Feature-extraction_Classification which is a very complex pipeline and the accuracy is dependent on the type of image processing and feature extraction techniques used [16]. It can achieve a good result, but previously found features are very sensitive. These methods relying on hand-crafted features don't generalize well. Most of these segmentation methods will miss the extraction of the area with the infection. The results of this approach are given in Figure 2.

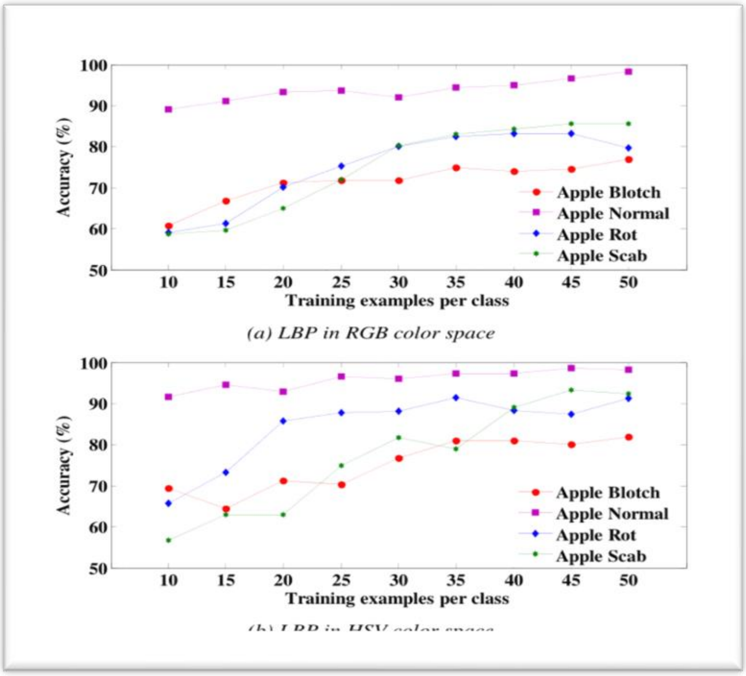

Figure 2.1: Accuracy of SVM classifier for apple diseases.

Table 2.1: Different works on fruit’s diseases.

Ref	Diseases	Description	Features	Classification Technique and Accuracy
[9]	(apple scab,apple rot,banana sigotka,banana leafspot,banana diamond leaf	proposes a novel classification architecture by using a pre-trained deep learning network combine with traditional classifiers	Alex-Net,Resnet50, GoogleNet architecture, and VGG	classified by traditional classifiers which are SVM, ELM and KNN 97.86%
[10]	Citrus canker, Black spot citrus scab	On quarantine illness was ,a association of a feature selection method and a classifier trained.	Includes shape ,color,texture.	Naïve, Bayes
[6]	Papaya Black spot	Proposes a general pipeline with Bicubic interpolation, Contrast Enhancement, k-means clustering	Texture and color by co-occurrence Matrix ,Statistical Features	By using SVM as a classifier the classification for normal and affected papaya fruits are 90%
[8]	Apple(scab,rot, blotch)	Based their technique on K-means clustering; Global Color Histogram (GCH), Color Coherence Vector (CCV), Local Binary Pattern (LBP)	Texture and color	The classification was achieved by MSVM
[16]	Pomegranate(anthracnose,Downey mildew)e	Canny edge detector, median filter, gray level co-occurrence matrix,	Color, texture	Average accuracies are 91.3%and 86.71% using GLCM and Glrm features.
[17]	Early scorch, , late scorch, sun bun	CCM AND SVM	Texture and color	Evaluation of the texture statistics and the accuracy was 94%.

2.3. Deep Learning and Agriculture

Deep Learning is a novel domain of research in ML that was introduced with one aim, bringing ML closer to it’s main goal: AI (artificial intelligence). The creation of these algorithm was inspired by the shape and function of the human brain. They have several grades of figuration to pattern the complicated relationships between the data. Deep Learning has been designed with Artificial Neural Network in mind and is designed to handle large areas and amounts of data by adding layers to the network. A deep learning model has the ability to extract raw data characteristics through many processing layers consisting of several linear and non-linear transformations and to learn more about these features step-by-step with minimal human intervention [19] [20] [21] [22]. Over the last five years, deep learning has gone from a niche market where only a handful of researchers have been interested, in the area most popular with researchers. Research related to deep learning and agriculture now appears in top journals like science.

2.4. The Convolutional Neural Network in Place of the Classical Approach

As described in Figure 2.2, the classical approach is defined as an algorithm that extracts the feature from an image, and use the extracted feature for classification. This approach as we said previously is very good, in terms of accuracy and performance. But during these last few years, with the use of convolution neural networks, a special deep learning network specializes in image processing and pattern recognition being multilayer supervised networks, can automatically learn entities from datasets. The development of this algorithm has been influenced by the failures of other algorithms to learn a high level of representation and to generalize a form of artificial intelligence [23]. For example, other machine learning algorithms failed to have adequate results in the artificial intelligence field such as object recognition, image segmentation or voice recognition. CNNs have achieved peak performance in almost all-important image classification tasks in the last years. They have the noble distinction of performing both extraction and classifications functions in the same architecture. Learning features automatically is one of the biggest benefits of CNNs and more without a need of segmentation method [16]. It has been proposed to use various CNN architectures for object recognition for example. LeNet, AlexNet, GoogLeNet etc. This deep learning architecture has been incredibly successful in plant disease detection studies. Deep learning is a very exciting subject indeed, [24] propose for the classification of banana leaves including healthy (1643 images) and unhealthy (2057images) fruits, the LeNet architectures as a convolutional neural network. They applied a technique called transfer learning. In [25] a dataset of 54306 images was used, they applied AlexNet and GoogleNet models based on CNNs to detect and identify 26 diseases, the final accuracy was 99.35%. For the identification of soybean plant Disease [16] uses a CNN architecture that consists of convolutional layers, where each of them is accompanied by a max-pooling layer and the fully connected MlP was used at the final layer. They achieved 99.32% of accuracy. To conclude this section, we can say that the main advantage of deep neural networks is that, they are most robust and automated. The deep neural networks are powerful techniques that can learn features automatically, it’s easier to generalize and use it in different and related problems.

2.5. Transfer Learning

In practice, according to Yosinski et al. [26], training a ConvNet model from the beginning (with initializations) is not recommended because, training a ConvNet model requires a lot of data and takes a lot of time. On the other hand, it is more common to use ConvNet models already in training and readapt them for the problem, this is called learning transfer. It is about transferring the learning of a model treating a problem towards another type of problem. There are two types of transfer learning.

- Extraction of ConvNet variables: here, the ConvNet is used as an extractor [27], ie. a vector is extracted from a certain layer of the model without modifying its structure or its weight and the previously extracted vector is used for a new task.
- The fine-tuning of the ConvNet model [28]: here, the new ConvNet is initialized with the weights and structure the model of pre-training to be utilized. The structure of the model of pre-training is slightly changed for the new task and finally, the new model is trained for the new task.

For their reliability and performance, these machine learning systems (ML) have been a great success because of the difficulty of the subject, recognition of the object has gained notoriety in the fields of research. In recent decades, the recognition of shapes, the classification of data, have become a great concern.

Throughout this chapter, we introduced the state of the arts as well as some concept of image classification, then we saw applications of deep learning in the field of agricultural imagery. Lastly, we did a thorough study of the methods of transfer learning[12]. On the other hand, SVM is considered as one of the strongest and robust algorithms in machine learning (ML) created by Vapnik [13] and have been a well–known approach exploited in many domains [14, 15, 16], like pattern recognition, classification, and image processing.

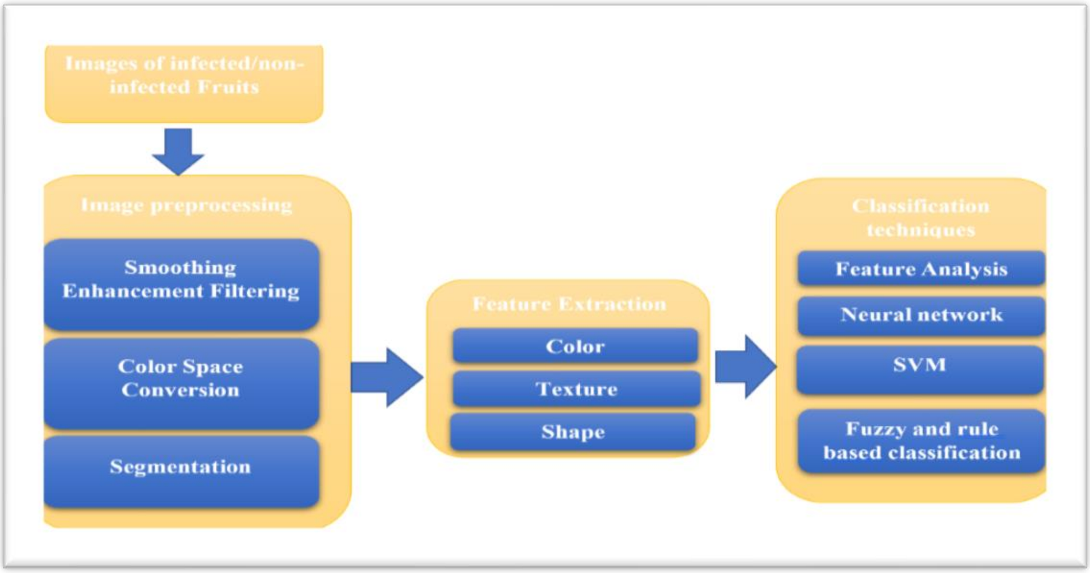

Figure 2.2: Pipeline follow by the classical approach.

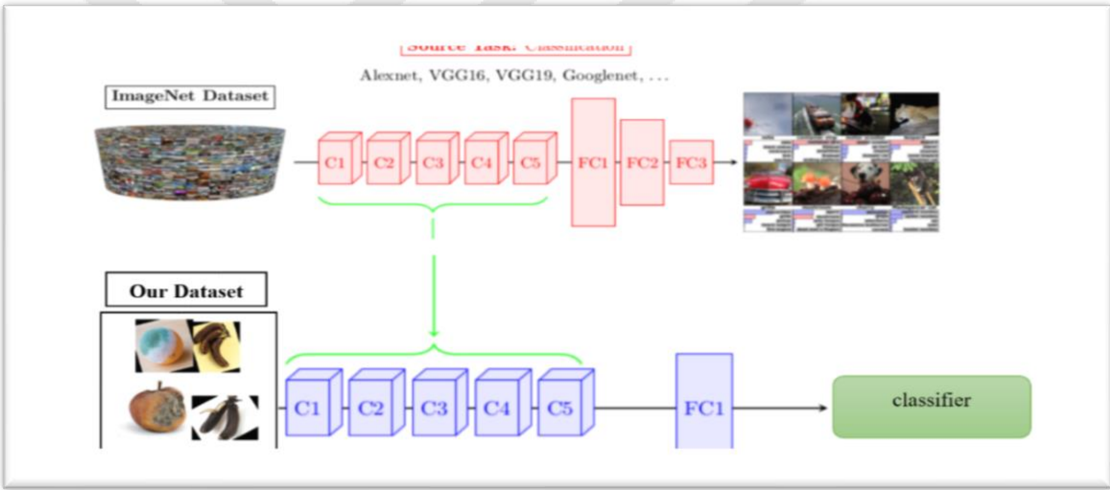

Figure 2.3: Pipeline of transfer learning.

3. RESEARCH PROCEDURE AND BACKGROUND INFORMATION

The research took place in three stages: (a) To learn more about the convolutional neural network in order to decide which CNN architecture to use. (b) Learn more about the different color spaces, how to go from one color space to another and (c) Define the Architecture of the model we want to propose.

3.1. Background Information

There are many variants of deep architecture. Most of them are derived from the original parenting architectures and several of them are obtained from the initial parenting architectures. Convolutional neural networks are characterized by an automatic extractor of features and a classifier that can be trained. The interrogation of convolution and sub-sampling layers are different. This difference lies in their architecture but, before we go further it is better to explain how Convolutional Neural Network works.

3.1.1. Neural Network

The artificial neural network is an automatic learning algorithm that uses certain aspects of animal neurons. It is mathematically modeled by a network in a graph, more or less complex, hierarchized in the form of layers whose elementary nodes are called neurons. It is generally composed of three types of layers: the input layer, hidden layer (s) and the output layer. The input layer is composed of neurons that correspond to the characteristics of the input data represented by a multidimensional grid (for example the pixel matrix of the image or the vector shape of the data). The output layer represents the results of the task assigned to the network. Take the example of a classification of 1,000 classes, the 1,000 neurons of the output layer represent the probability or score for each class. The hidden layers are the intermediate layers between the input and the output. The entire network thus formed is generally seen as a black box.

Theorem 3.1. Hornik's theorem [36, 15, 29, 16]: A neural network with a hidden 1-layer has the universal approximation property, it can approach any continuous function with arbitrary precision, provided there are enough neurons on it's hidden layer, as any continuous function can be approximated with a network of 1-layer neurons (see Hornik's theorem), the idea of deep learning comes from increasing the number of hidden layers of the network. Hence the deep term, to be able to learn a higher level of abstraction. [5]

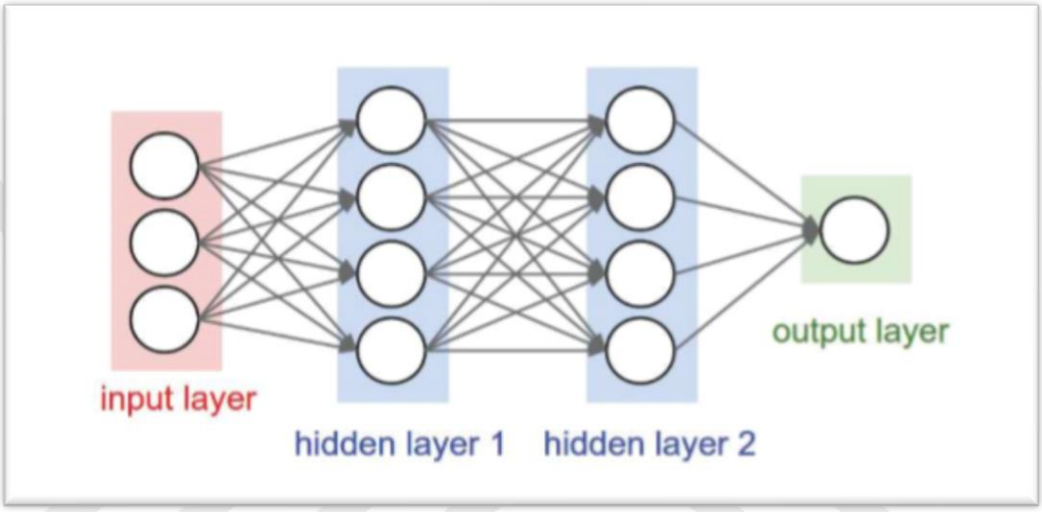

Figure 3.1: Neural network or neural networks with two hidden layers; fully connected neural network.

A neural network can be qualified deep if the number of hidden layers is greater than or equal to 2 (see for example Figs. 3.4). In the next subsection, we will detail it's operation and the other types of neural networks that we have exploited for this research.

3.1.2. Activations Functions

A biological neuron is characterized by(see Figure 3.2):

- Synapse: The point of connection with other neurons,
- Dendrites: The points of entry of the neuron,
- Axon: The exit point,
- The kernel: Which activates the outputs according to the input stimulations.

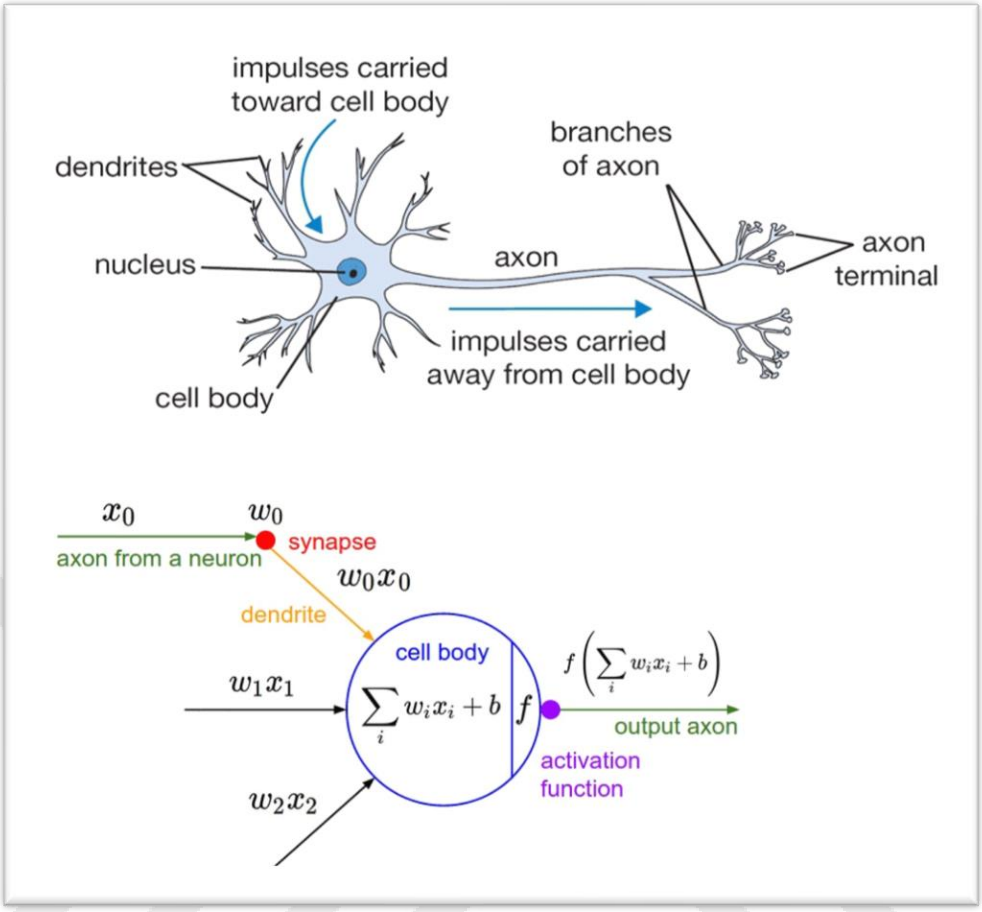

Figure 3.2: Comparison between an animal neuron and the mathematical model of an artificial neuron.

In biological networks, the input data comes from another neuron that passes into the dendrite, the cell body does the calculations and the result comes out through the axon. The functioning of an artificial neuron is traced to that of a biological neuron. The image is taken from [45]. By resemblance to a biological neuron, and an artificial neuron is defined by (see Figure 3.2):

- A weight vector w of the same size as the input signals,
- Input signals X = $x_1, \ldots, x_n$,
- An activation function g,
- The neuron calculates the dot product between the input signals and the weight vector by adding a bias. The result is applied in the activation function.

$$S = g\,(X.w\ + b)\ = g\,(\sum_{k=1}^{n} x_k\ *\ w_k\ +\ b) \tag{3.1}$$

S: the value or the internal state, or the internal score of the neuron. b: variable bias of the neuron. The activation function g is used to introduce a non-linear operation after the dot product operation (see Equation 3.1). This non-linearity makes it possible to have different variations of the internal state on an object of the same class. The most used activation functions are: Sigmoid function, commonly used in output layers, not in the hidden.

$$\sigma(x)\ =\ \frac{1}{1\ +\ e^{x}} \tag{3.2}$$

Tanh function, this function is similar to (8), used in the hidden layer, not in the output. gives us a value between -1 and 1. As we want to classify, it is important to have a function with which we can obtain values between 0 and 1.

$$\tanh(h)\ =\ \frac{e^{z}\ - e^{-z}}{e^{z\ + e^{-z}}} \tag{3.3}$$

Rectified-Linear unit (ReLU) : relu(x) = max(0,x)

Tanh, sigmoid and ReLU models are well suited to optimization algorithms such as stochastic gradient descent, as they are differentiable. But currently, ReLU is the most used in deep neural networks because of a problem called "vanishing gradient".Like the other activation functions (sigmoid, tanh) produce values between 0 and 1 the gradient will tend to 0 because deep networks (multiple layers) are available ReLU allows to mitigate the "vanishing gradient" because it’s output value is invariant if x is > 0. In addition, the calculation of exponentials present in the sigmoid functions and tanh is slow compared to the calculation of the max (9) in ReLU which is therefore preferred.

To predict from a model of artificial neural networks (see Equation 3.3), we cross the network from the input layer and then successively calculate the values of the internal states of each neuron in the hidden layers, up to the output layer. Then, the internal values of the output layer are applied to what is called an entropy function. By definition, an entropy function is a mathematical function used to transform the internal state values of the output layer neurons into probabilities. Prediction is the class corresponding to the neuron that has a high probability. Generally, we use softmax entropy or sigmoid is used with as respective softmax loss function or sigmoid entropy loss.

3.2. Convolutional neural networks

The convolutional neural network (convolutional neural network or ConvNet or CNN) is very similar to a neural network of the type mentioned above. ConvNets is widely used in the field of computer vision if we only mention imageNet and Pascal VOC, which are problems of detecting and classifying objects in an image. They are a type of specialized neural network for processing data with a grid-like topology. Also they are very successful in facial recognition, the detection of objects that are widely used in robots and automatic cars. Basically, everything about computer vision and images. Moreover one can use the convNet in all the problems having as input a matrix

Convolutional networks have had considerable success in practical applications. They specialize in the processing of matrix and signal data. "Convolutional neural network" refers to the use of a mathematical operation, namely the convolution operation. which is a particular linear operation. A CNN is simply a network that does not use matrix multiplication in at least one of their layers but uses the convolution of the layers instead. Note that the term "convolutional" comes from the matrix convolution operation used in signal processing. In convNet, two new layer types have been added to the network: the convolutional layer and the pooling layer. We describe them in the following sections. They have wide applications in image and video recognition, recommendation systems [23] and natural language processing [29]–[31]

3.2.1. The Convolution Operation

In its most general form, convolution is an operation on two functions of the real argument. Let’s talk about the motivation behind convolution.

$$c(t) = \int x(a)w(t-a)da \tag{3.4}$$

This operation is named convolution. Which is usually designated with an asterisk.

$$c(t) = (x * w)t \tag{3.5}$$

If we assume that x and w are integers, we can define the discrete convolution:

$$c(t) = (x * w)t = \sum_{a=-\infty}^{\infty} x(a)w(t-a) \tag{3.6}$$

In the terminology of the convolutional network, the first argument (in this instance, the x function) of the convolution is frequently called input and the second element (in this case the function w) as the kernel. The output is known as a feature map.

3.2.2. Convolution layer

In the convolutional layer, instead of creating a scalar product between the internal values and the weights of each neuron (equation 3.6), a convolution product is applied. This convolution product makes it possible to extract specific characters in the signal or image processed. In a CNN, a succession of convolutional filters is applied to gradually makes the extraction of the characteristic information of the image. The convolution operation is shown in Figure 3.3. It is calculated by equation 9 or S is the resulting convolution product [20], [24], [32].

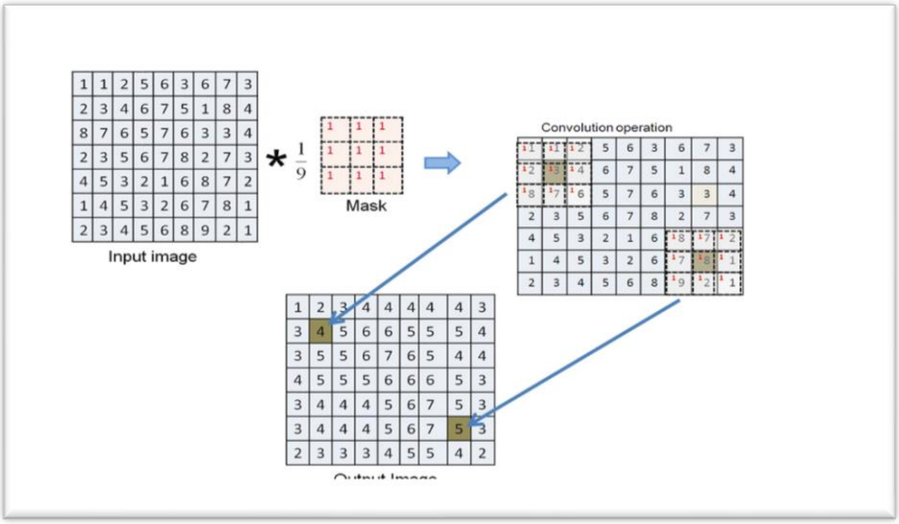

Figure 3.3: Illustration of the pooling layer, with the method of max pooling.

3.2.3. Pooling layer

The pooling layer is a kind of non-linear sub-sampling. It is used to reduce the spatial dimension in a ConvNet network. This is a layer that takes in each "feature map" [1]and simplifies it. For example, the "max-pooling" takes a region and gives the maximum output of this region. (figure 3.4).

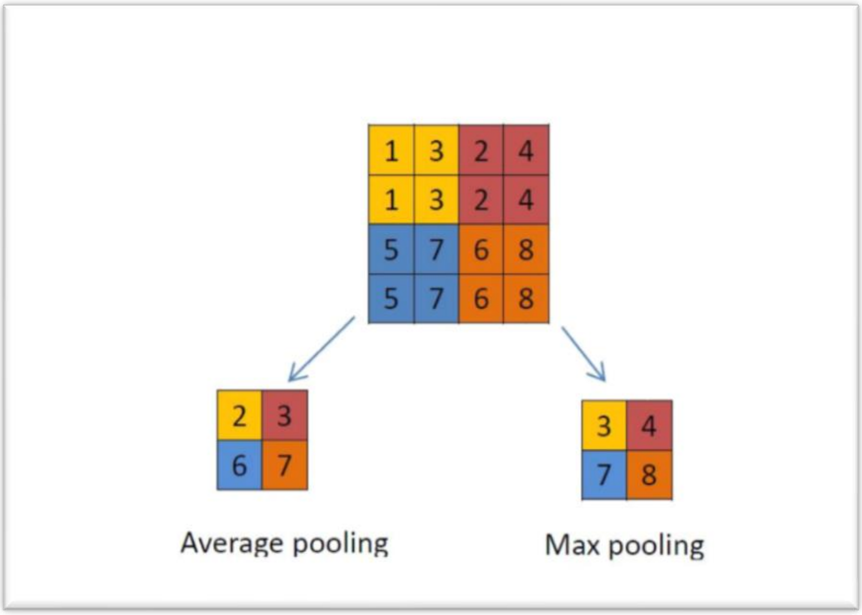

Figure 3.4: (on the left) Average pooling: The yellow box: (1+ 3+ 1+ 3) / 4 = 2. (on the right) Max pooling: example of the blue box: max (5, 7, 5, 7) =7.

3.2.4. Flattening

In the last step of the extraction part, flattening is simply to put end to end all the images (matrices) that we have to make a (long) vector. The pixels (actually they are no longer images or pixels, but matrices of numbers, so the pixels are those numbers) are retrieved line by line and added to the final vector.

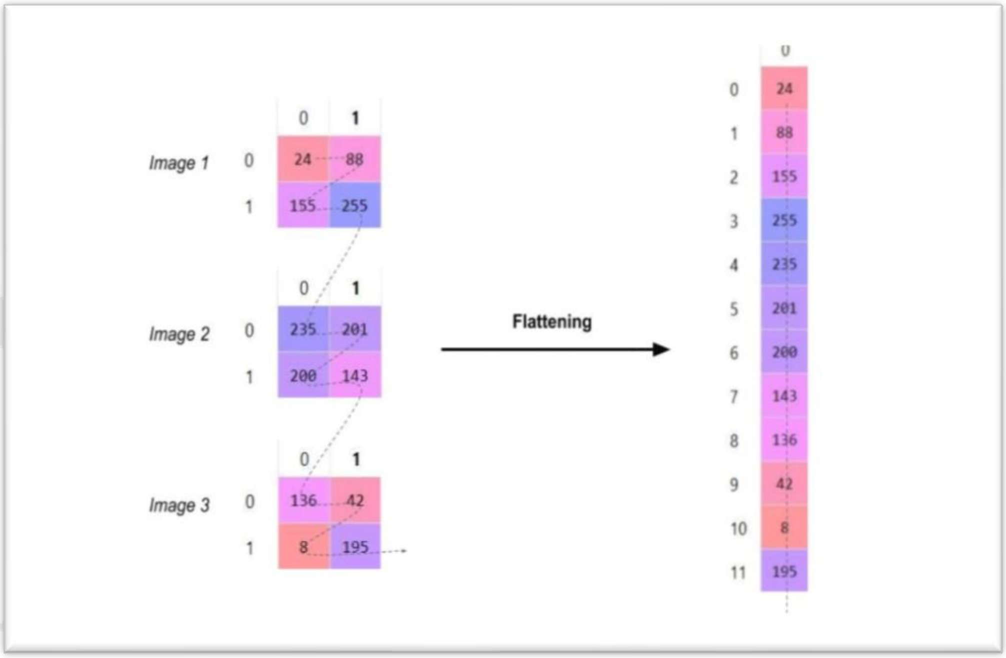

Figure 3.5: Final flattening of the final images of the filters + simplifications.

3.2.5. Fully Connected Layer

After the flattening step where we obtain a vector whose output will be the input of the fully connected layer. And the output of the fully connected will be the number of the classes we have.So the structure of the fully connected is described in Figure 3.6.

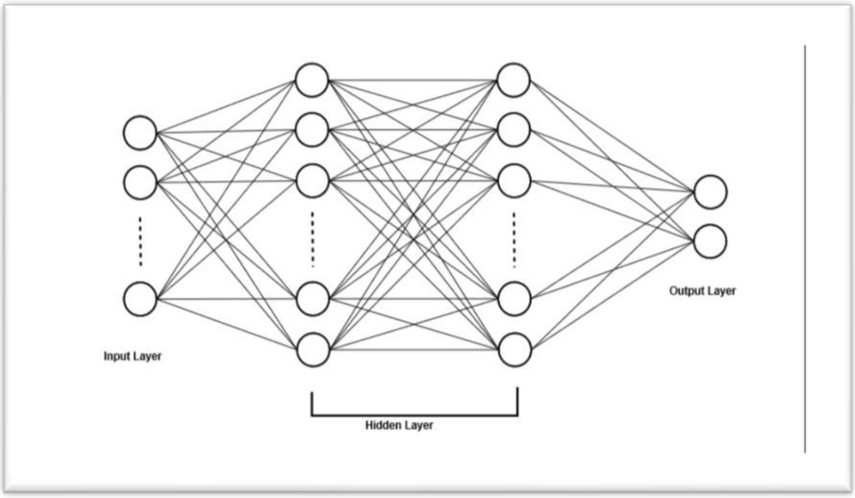

Figure 3.6: Fully-connected layer structure.

Input neurons, which send their value to all the neurons of the next layer. In the case of CNN, it will be the value of a precise pixel for each neuron (simply, following the order of the vector). Hidden neurons, organized in layers, which will send the sum of the signals they receive (weighted by the "importance" of their connection) to the neurons of the next layer. Output neurons, which receive the sum of the weighted signals of the last hidden layer.

3.3. The Architectural Prototype of a Convolutional Neural Network

After having all the necessary tools to understand the architecture of a convolutional neural network. Let’s see the different network that we found in the literature, whose effectiveness varies according to the tasks. These included:

- LeNet (the simplest, to discover the CNN),
- AlexNet (launched in 2012),
- ZFNet (improved AlexNet forgotten),
- GoogLeNet (parent of the famous concept Inception, an image recognition algorithm with other subtleties, just like YOLO, MobileNet, etc.),
- VGGNet (always very used and powerful),
- ResNet (same, even if it moves away from a traditional CNN).

3.3.1. Architecture of the Alex-Net Network

One of the network that has been chosen in this thesis is the Alex-Net Introduced in 2012 by Alex Krizhevsky and having an architecture similar to LeNet, but being deeper, larger and having convoluted layers stacked on top of each other. The architecture of this network consists of 8 layers, consist of five convolutional layers, 2 fully connected to the hidden layers and a fully connected output layer. The shape of the convolution window in the first layer, is 11 × 11. ImageNet's data objects are ten times higher than those of MNIST. So, they tend to occupy more pixels therefore, a larger convolution window is needed to capture the object. The size of the second convolution window is reduced to 3 x 3 after it has been reduced by 5 x 5. In addition, maximum pooling layers with a window shape of 3 × 3 and a stride of 2 are added to the network after the five convolutional layers. This network also used the replay activation feature. Figure 3.7 describes the architecture of the different parameters of the Alex model.

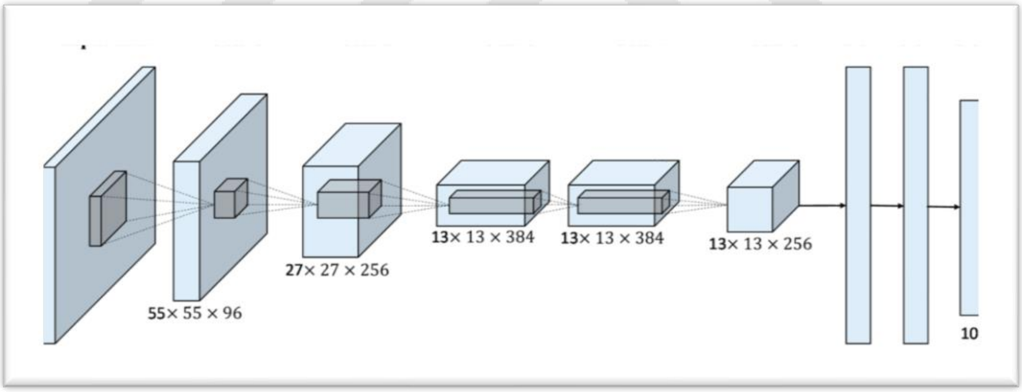

Figure 3.7: Alex-Net Architecture.

3.3.2. The architecture of the custom CNN

The second model is the one we have decided to develop ourselves.So we will give now the details of the new architecture. We would like to point out that the parameters were selected not by experience but empirically because, often we try several architectures before finding the best. For the construction of that network, we wanted something simple so the custom Cnns is composed of 3 convolution layers.

The first arguments of each convolution layer are the filters. In the custom network, we set the filters to 32,64 and 64. So 32 for the first block and 64 for the second and third block. Each of these filters has an identical size 3 by 3, then we apply the activation function "relu" for all the activation layers therefore, for improving the speed, the performance and the stability of the network we use the batch normalization parameters in the three layers. 'Padding' parameter is set as 'same' which is different in the Alex-net network which is set to 'valid'.

The MaxPooling is used and the pool size is 2 by 2. The pooling layer is set before using the Dropout parameter in each convolution block. the dropout is set at 0.25 value. One fully connected layer is defined with a Dense class followed by a 'relu' activation function, a batch normalization and a dropout parameter and then the model will end with a 'softmax' which will give the probability of each class. The summarization of the network will be given in the following part. Figure 3.8. Describe the architecture of the custom network.

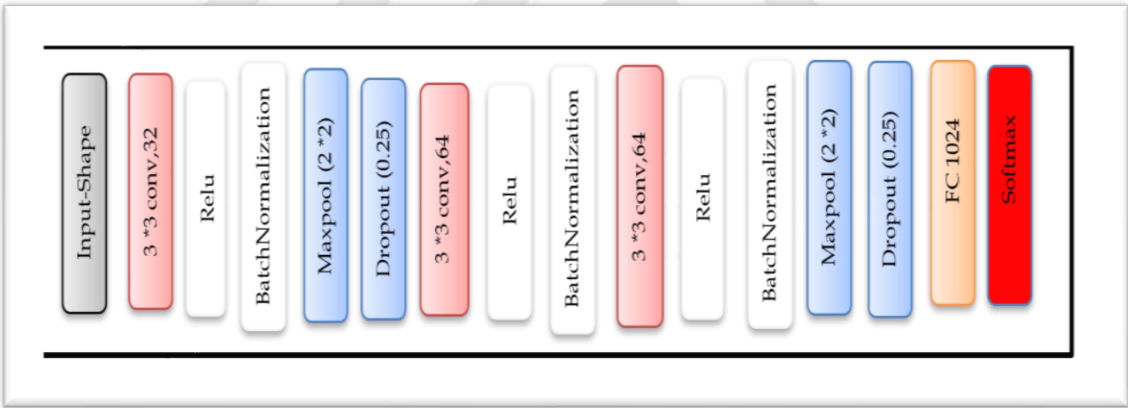

Figure 3.8: Custom Cnn Architecture.

3.4. Color Spaces

A color space is a defined color gamut, they allow us to define the color in both analog and digital ways. it can also be considered as an abstraction of a mathematical model. There are several color spaces, but the best known is the RGB color space. In this thesis, the chosen color spaces are RGB, HSV, LAB.

3.4.1. RGB

This color space is represented by three components R for red, G for green and B for blue, each color has 8 bits, which represents 24 bits in total. each color space has a value ranging from 0 to 255. It’s the most color images taken for the classification in computer vision. It’s considered as an additive color space.

As we said, in the introduction we want to see the impact that changes (transformation from RGB images) can have on the classification accuracy. At the beginning of our work, as we resize the input image, we also preprocessed the input images by converting them into multiple color spaces at the same time. The most recent models do not perform color space transformation in the image but perform their classification in RGB format.

3.4.2. HSV

Composed of three components whose hue corresponds to the color shade and where that color is in the color spectrum. Red, yellow, and purple are words that describe the hue. Figure 3.9,illustrates the range of hues, H, as a circle represented by values from 0 to 360. Saturation to the shade, the saturation describes how pure is the hue compared to a white reference, it is a percentage that ranges from 0 to 255. and value to brightness as the saturation the value has range 0 to 255. In conclusion the HSV was defined with reference to RGB and CMYK models. It is considered the closest model to the perception of human color.

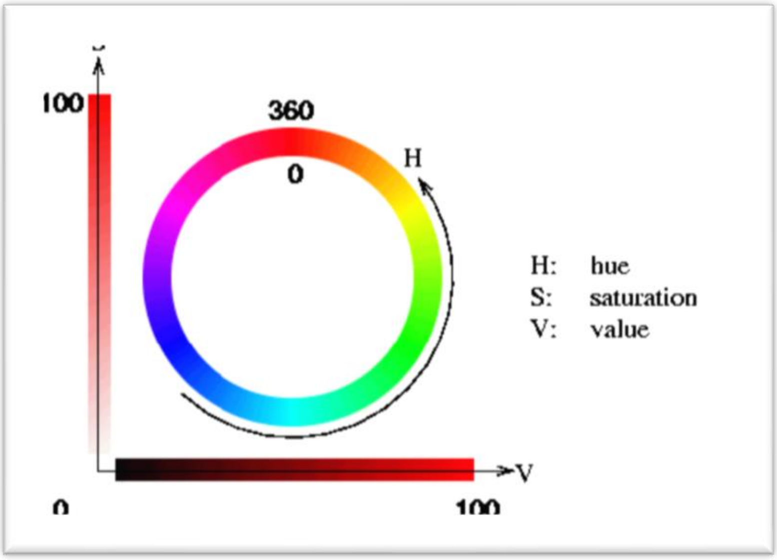

Figure 3.9: Hue, Saturation, and Value.

- Operations of the passage of RGB and HSV spaces.

After explaining the HSV color space, Its composition etc. Let Now see how we can transform the RGB to HSV. Its transformation is illustrated in (3.7)-(3.12).

$$R' = R/255, G' = G/255, B' = B/255 \tag{3.7}$$

$$Cmax = \max(R', G', B'), Cmin = \min(R', G', B') \tag{3.8}$$

$$\Delta = Cmax - Cmin \tag{3.9}$$

$$H = \begin{cases} \text{With} \quad 0° & , \Delta = 0 \\ 60°\left(\dfrac{G' - B'}{\Delta} \bmod 6\right) & , Cmax = R' \\ 60°\left(\dfrac{b' - R'}{\Delta} + 2\right) & , Cmax = G' \\ 60°\left(\dfrac{R' - G'}{\Delta} + 4\right) & , Cmax = B' \end{cases} \tag{3.10}$$

$$S = \begin{cases} 0 & , Cmax = 0 \\ \dfrac{\Delta}{Cmax} & , Cmax = 0 \end{cases} \tag{3.11}$$

$$V = Cmax \tag{3.12}$$

3.4.3. CIELAB

The CIELAB model (CIE LAB), adopted by the CIE in 1976 is an alternative to the XYZ model. It models the human vision and is independent of any material. The colors are defined by three values: Luminance (luminance) coded in percentages a and b correspond to the color information (chrominance) where, the color is defined from a mixture of green to magenta (a) and a mixture of blue to yellow (b). Values range from -120 to +120 for a and b.

- Operations of the passage of RGB and Lab spaces.

  The first step is to switch from RGB components to XYZ components. This is done using a conversion matrix.

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{bmatrix} 0.41 & 0.36 & 0.18 \\ 0.21 & 0.72 & 0.07 \\ 0.02 & 0.12 & 0.95 \end{bmatrix} \cdot \begin{pmatrix} R \\ G \\ B \end{pmatrix} \tag{3.13}$$

  Then, it's about moving from the XYZ space to the LAB space. The following conversion formulas are then used:

For $\frac{y}{y_n} > 0.008856$

For $\frac{y}{y_n} > 0.008856$

$$L = 116(\frac{y}{Y_n})^{\frac{1}{3}} - 16$$

$$L = 903.3(\frac{y}{y_n})$$

$$a = 500(f(\frac{X}{X_n}) - f(\frac{y}{y_n})) \tag{3.14}$$

$$b = 200(f(\frac{y}{y_n}) - f(\frac{Z}{Z_n}))$$

For $t > 0.008856 \quad f(t) = \sqrt[3]{t}$

where

For $t > 0.008856 \quad f(t) = 7.7787 + \frac{16}{116}$

Xn, Yn and Zn correspond to the white described in the XYZ space. They are obtained for RGB = (255,255,255). When it comes to work on the colors of an image, as in the case of a quantization, two colors that are close in the color space (in the sense of the Euclidean distance most often), may appear quite different for the eye, which is the case for the RGB color space. On the other hand, in the LAB space which is uniform, two colors close in distance are also for the eye. In this work, only three of the most popular color spaces have been taken into account. The final architecture will be explained in the next section.

3.5. The architecture of the proposed model

After defining the different architectures of CNNs that will be used, we then define the final architecture of the proposed model.

In the proposed architecture, the part that will be different from the works that have been already done in the field of fruit classification is illustrated as follows: the RGB input image will be simultaneously converted into two other color spaces and then, from each conversion, the RGB input image will be concatenated with HSV and LAB color spaces to form a new input for our networks. The general architecture of the model is shown in Figure 3.10.

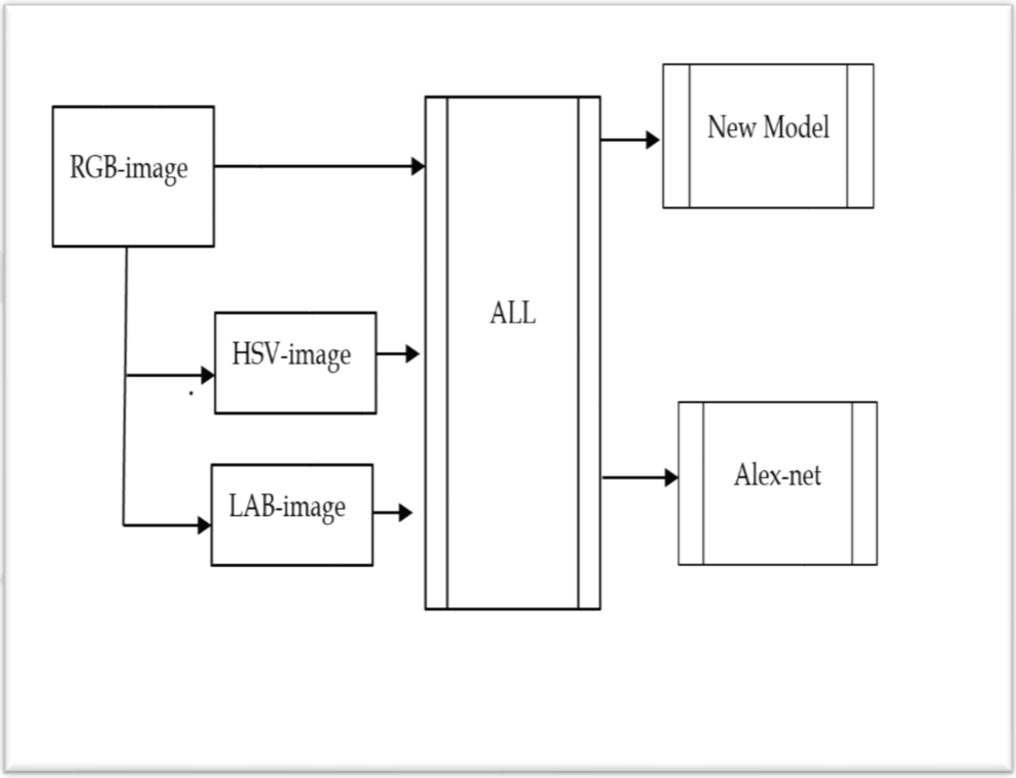

Figure 3.10: Model Architecture.

4. IMPLEMENTATION

In this chapter, we presented the implementation of the proposed approaches, by presenting the tools used and the technological choices. The different processes in the implementation of the models are discussed in detail.

4.1. Workflow

Several steps were performed throughout the process. The most important is the configuration and installation of the environment and packages. Then, we built our networks after the environment was done. At this stage, before feeding our data for training, we applied image processing methods to prepare our data. Finally, in the final epic, we evaluate the results obtained during the training phase. The different steps are detailed in the figure 4.1.

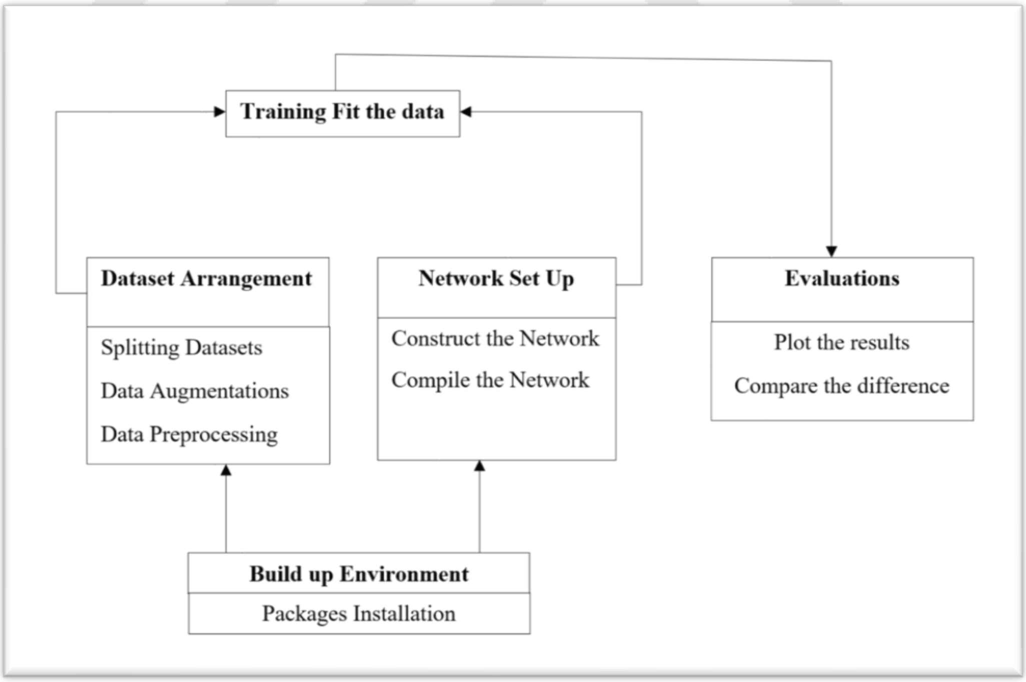

Figure 4.1: General workflow.

4.1.1. Pre-request

This framework will help us to build and train our CNNs. The required packages are:

- Numpy and Scipy
- pandas
- OpenCv_python == 3.4.2.17
- pillow
- Tensorflow
- Keras

For the installation of all these packages we used the Anaconda distribution. our editor/ is spyder. Numpy: is a library of Python to make matrix and array operations in multi-dimensions also to manipulate vectors and interact with TensorFlow, our Deep Learning framework. This library is very widely used. Keras: the deep learning framework. The import of the Tensorflow backend is done automatically by Keras.

4.1.2. Hardware

Deep learning is a domain with intense computing requirements and the availability of resources (especially in GPU) dedicated to this task will fundamentally influence the user experience because, without it's resources, it will take too much time to learn from his mistakes which can be discouraging. The experiments were all performed on a machine that offers acceptable performance and features:

Table 4.1: Hardware.

CPU	INTEL CORE İ7-7500U (2.90 GHz)
RAM	8GB

But as the code will be put to the disponibility of everyone we have configured the code in a way that can be used in a computer with better resources than ours (With GPU) and can be executed in windows and Ubuntu.

4.2. Datasets

To perform this experimentation, we utilized two datasets. The first one from kaggle called “fruits fresh and rotten for classification” is a public dataset with six classes, and the second one that we gathered ourselves consisting of only apples. Both of the datasets contain fresh and rotten fruits. The details of the datasets are given in Table 4.2.

Table 4.2: Fruits fresh and rotten for classification dataset.

Kaggle-dataset	**Type of fruits**	**Train**	**test**
1	fresh apples	1693	395
2	fresh banana	1581	381
3	fresh oranges	1466	388
4	rotten apples	2342	601
5	rotten banana	2224	530
6	rotten oranges	1595	403
Total of images:	**11375**		

Table 4.3: New-Dataset.

New-dataset		**train**	**test**
1	fresh apples	373	167
2	rotten apples	88	38
Total of images:	**666**		

For completing the experiments, the images are separated into two folders, the training set and the testing set. It was with a ratio of 0.8 that the separation was done. 80% of the images were dedicated to the training set and 20% to the test set. In the files, there are sub-files with the description of the category names. For us, the experimentation was done by combining both datasets, as the new dataset was not enough, we combine the kaggle-dataset and the new-dataset. Then, we obtained the new repartition in Table 4.3.

Table 4.4: Concatenated Dataset.

Combined-dataset	Classes	train	test
1	fresh fruits	6249	1572
2	rotten fruits	5113	1331
Total of image:	14265	11362	2903

For the different experiments, the names of the categories were used, as the class names for the different models.

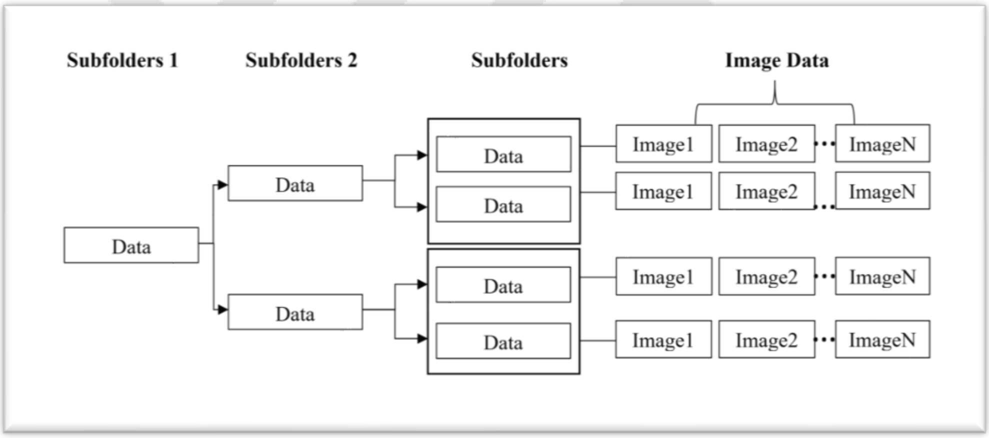

Figure 4.2: Data arrangement.

4.2.1. Augmentation Data

After studying the dataset more closely, we realized that the kaggle-dataset had already been increased by 15, 45 and 60 degrees per rotation. Thus, we did not apply any further data augmentation in the dataset.

4.2.2. Data Preprocessing

Before injecting the data to CNN some treatments were required especially due to the fact that we used multiple colors spaces. these treatments can have an impact on the results we need to obtain.

To start with, a load_data_into_batch_generators function was made so that the learning data could be preprocessed. In this function the images are downloaded and then split into training and data validation with the help of train_test_split function from the sklearn model_selection package. Then in the Create_batch_generator function the images are resized into (224,224,3) meaning all the images will have the format (224,224) that the format required for Alex-Net and the option antialiasing was used in order to make the image more clear, but also to preserves the best quality of the images. In the following, we define the lot size over a reasonable whole number, with the size of big data sets, when we can not import the data into the network as a whole, then we apply it. As the images loaded are in RGB color space which the initial color space, the images are converted to HSV and LAB color spaces. To convert RGB images to HSV and LAB color space we use the cvtColorfunction in OpenCV package. For the HSV color space, this parameter cv2.COLOR_RGB2HSV was used and for the LAB color space cv2.COLOR_RGB2LAB was used. After each conversion and concatenation of the colors, the matrices were divided by 255 in order to normalize the values. After each conversion the three color spaces were concatenating as we can see in the capture below.

```
def create_batch_generator(data,
                           split_dir,
                           shuffle=True,
                           batch_size=config.batch_size,
                           base_path=config.base_path,
                           input_shape=(224, 224, 3),
                           color_space='rgb'):
    data['x'] = config.base_path + '/'
                + split_dir + "/"
                + data['label'] + "/"
                + data['filename']
    data_x = data['x'].values
    n = len(data_x)
    data_y1 = pd.get_dummies(data['label'].values).values
    _image_shape = (input_shape[0], input_shape[1])
    while True:
        batch_start = 0
        batch_end = batch_size
        indexes = np.arange(len(range(0, n)))
        if shuffle:
            np.random.shuffle(indexes)
        while batch_start < n:
            limit = min(batch_end, n)
            index = indexes[batch_start:limit]
            batch_y1 = [data_y1[i] for i in index]
            batch_x = [data_x[i] for i in index]
            batch_images_x = list()
            batch_images_x_hsv = list()
            batch_images_x_lab = list()
            image = None
            for image_name in batch_x:
                image_rgb = load_img(os.path.join(base_path,
                                                  image_name)).resize
                                                  (_image_shape,
                                                  Image.ANTIALIAS)

                if color_space == 'hsv':
                    image = cv2.cvtColor(np.array(image_rgb),
                                         cv2.COLOR_RGB2HSV)
                elif color space == 'lab':
                    # L in [0, 100]
                    # A in [-86.185, 98.254]
                    # B in [-107.863, 94.482]
                    image = cv2.cvtColor(np.array(image_rgb),
                                         cv2.COLOR_RGB2LAB)
                elif color_space == 'rgb':
                    image = image_rgb
                elif color_space == 'all':
                    image = image_rgb
                    image_hsv = cv2.cvtColor(np.array(image_rgb),
                                             cv2.COLOR_RGB2HSV)
                    image_lab = cv2.cvtColor(np.array(image_rgb),
                                             cv2.COLOR_RGB2LAB)
                    batch_images_x_lab.append(img_to_array(image_lab))
                    batch_images_x_hsv.append(img_to_array(image_hsv))
                batch_images_x.append(img_to_array(image))

            batch_images_x = np.array(batch_images_x)
            if color_space == 'all':
                batch_images_x_hsv = np.array(batch_images_x_hsv)
                batch_images_x_lab = np.array(batch_images_x_lab)

            if color_space in ['rgb', 'hsv']:
                batch_images_x /= 255
            # TODO Add normalization for LAB colors

            if color_space == 'all':
                batch_images_x_hsv /= 255
                batch_images_x /= 255
                batch_images_x = np.concatenate((batch_images_x,
                                                 batch_images_x_hsv,
                                                 batch_images_x_lab),
                                                 axis=-1)
            batch_images_y1 = np.array([y for y in batch_y1])

            yield (batch_images_x, batch_images_y1)  # for single outp
            batch_start += batch_size
            batch_end += batch_size
```

Figure 4.3: Data processing.

4.3. Training the CNN

After,the preparation of data, the labeled dataset can now be load and feed the data to the models. the proportions of the division of the dataset are given in the section below.

4.3.1. Construction of CNN and Alex-Net Network

- Alex-net

Layer (type)	Output Shape	Param #
conv2d_6 (Conv2D)	(None, 54, 54, 96)	34944
activation_10 (Activation)	(None, 54, 54, 96)	0
max_pooling2d_4 (MaxPooling2	(None, 27, 27, 96)	0
conv2d_7 (Conv2D)	(None, 17, 17, 256)	2973952
activation_11 (Activation)	(None, 17, 17, 256)	0
max_pooling2d_5 (MaxPooling2	(None, 8, 8, 256)	0
conv2d_8 (Conv2D)	(None, 6, 6, 384)	885120
activation_12 (Activation)	(None, 6, 6, 384)	0
conv2d_9 (Conv2D)	(None, 4, 4, 384)	1327488
activation_13 (Activation)	(None, 4, 4, 384)	0
conv2d_10 (Conv2D)	(None, 2, 2, 256)	884992
activation_14 (Activation)	(None, 2, 2, 256)	0
max_pooling2d_6 (MaxPooling2	(None, 1, 1, 256)	0
flatten_2 (Flatten)	(None, 256)	0
dense_5 (Dense)	(None, 4096)	1052672
activation_15 (Activation)	(None, 4096)	0
dropout_4 (Dropout)	(None, 4096)	0
dense_6 (Dense)	(None, 4096)	16781312
activation_16 (Activation)	(None, 4096)	0
dropout_5 (Dropout)	(None, 4096)	0
dense_7 (Dense)	(None, 1000)	4097000
activation_17 (Activation)	(None, 1000)	0
dropout_6 (Dropout)	(None, 1000)	0
dense_8 (Dense)	(None, 2)	2002
activation_18 (Activation)	(None, 2)	0

Total params: 28,039,482
Trainable params: 28,039,482

Figure 4.4: Alex-Net network summary

- Custom CNN

```
ayer (type)                  Output Shape              Param #
=================================================================
onv2d_14 (Conv2D)            (None, 224, 224, 32)      896
_________________________________________________________________
ctivation_25 (Activation)    (None, 224, 224, 32)      0
_________________________________________________________________
ctivation_26 (Activation)    (None, 224, 224, 32)      0
_________________________________________________________________
atch_normalization_5 (Batch (None, 224, 224, 32)      128
_________________________________________________________________
ax_pooling2d_9 (MaxPooling2 (None, 74, 74, 32)        0
_________________________________________________________________
ropout_10 (Dropout)          (None, 74, 74, 32)        0
_________________________________________________________________
onv2d_15 (Conv2D)            (None, 74, 74, 64)        18496
_________________________________________________________________
ctivation_27 (Activation)    (None, 74, 74, 64)        0
_________________________________________________________________
atch_normalization_6 (Batch (None, 74, 74, 64)        256
_________________________________________________________________
onv2d_16 (Conv2D)            (None, 74, 74, 64)        36928
_________________________________________________________________
ctivation_28 (Activation)    (None, 74, 74, 64)        0
_________________________________________________________________
atch_normalization_7 (Batch (None, 74, 74, 64)        256
_________________________________________________________________
ax_pooling2d_10 (MaxPooling (None, 37, 37, 64)        0
_________________________________________________________________
ropout_11 (Dropout)          (None, 37, 37, 64)        0
_________________________________________________________________
latten_4 (Flatten)           (None, 87616)             0
_________________________________________________________________
ense_11 (Dense)              (None, 1024)              89719808
_________________________________________________________________
ctivation_29 (Activation)    (None, 1024)              0
_________________________________________________________________
atch_normalization_8 (Batch (None, 1024)              4096
_________________________________________________________________
ropout_12 (Dropout)          (None, 1024)              0
_________________________________________________________________
ense_12 (Dense)              (None, 2)                 2050
_________________________________________________________________
ctivation_30 (Activation)    (None, 2)                 0
=================================================================
otal params: 89,782,914
rainable params: 89,780,546
```

Figure 4.5: Custom-Cnn network summary.

Depending on the color space we used, the input shapes varied. In Figure 4.4 and Figure 4.5, we can see the summarization of the Alex-net and the custom-CNN with their output shapes and the parametres numbers. We will not enter deep in to the details because the architecture was explained in the previous section.

4.3.2. Compile the Network

As our models are defined and created, the compilation of its networks can take place by using the function compile_model in Figure 4.6, then the models can be compiled. In this function, parameters must be specified and we chose to define optimization and loss.

```
def compile_model(self, learn_rate=0.01):
    opt = SGD(lr=learn_rate, clipnorm=1.)

    self.cnn.compile(loss='categorical_crossentropy',
                     optimizer=opt,
                     metrics=['categorical_accuracy'])
```

Figure 4.6: Compile model example.

Logarithmic loss has been chosen, the loss assesses the accuracy of the CNN and allows it to be better adapted to the data if necessary, ‘categorical_crossentropy’ was used. Optimizer: it's an algorithm that will dictate how to update the CNN to decrease the loss in order to have better predictions. For the optimizer, stochastic gradient descent (SGD) was used, the SGD choose randomly a subset of training sample to update the parameters. Metrics: it's exactly like the loss, except that the metrics are not used by the CNN, unlike the loss that is used to update CNN variables via the optimizer. We used this time "accuracy", without this having any real importance for us.

4.3.3. Fit the Network

To train a model, just use the fit_generator() function which reads the data from a generator and the model is then trained batch by batch.

```
def train_model(model, batch_generators, steps, model_name, print_summary=False):
    _callbacks = utils.get_callbacks(model_name=model_name)
    _workers = utils.get_workers()
    if print_summary:
        model.cnn.summary()
    _ = model.cnn.fit_generator(generator=batch_generators['train'],
                                steps_per_epoch=steps['train'],
                                epochs=1000,
                                verbose=1,
                                max_queue_size=_workers * 2,
                                validation_data=batch_generators['validation'],
                                validation_steps=steps['validation'],
                                shuffle=True,
                                workers=_workers,
                                use_multiprocessing=True if _workers > 1 else False
                                initial_epoch=0,
                                callbacks=_callbacks)
    return model
```

Figure 4.7: Example code model.

Fit the network in one sentence means to feed the data into the constructed and compiled model. During 1000 epochs, the training process runned through the data. During the training process, we visualized that the training accuracy and the loss training keeping growing towards 100 percent but, the validation accuracy was trapped around a certain percentage, then we decided to stop the training once this situation occurs by using what is called “early-stopping”, therefore, in our case the training stops if the accuracy is not improving for consecutive 10 epochs because, we assume that it will not increase ahead too and don’t need to train further. However, an efficient way to avoid the over-fitting situation is by enlarging the training data.

4.4. Classical Methods

The convolutional neural network is a powerful model in terms of classification and image recognition but we also wanted to try what we call traditional methods and compare the different results. This process includes the part of image processing, segmentation, feature extraction and finally classification which are the similar steps used in CNNs models but we used different algorithms. The details of the different steps are given in Figure 4.8.

Step 1: Resize fruit images in size 224 × 224 based on deep networks with bilinear interpolation algorithm and improved image quality using a median filter algorithm.

Step 2: To follow the steps of the classical approach, we have chosen to add to our model the segmentation step. In order to detect the region of interest on the image.

Step 3: Extraction of Features by using different fully connected layers of the Alex-net model.

Step 4: Classification.

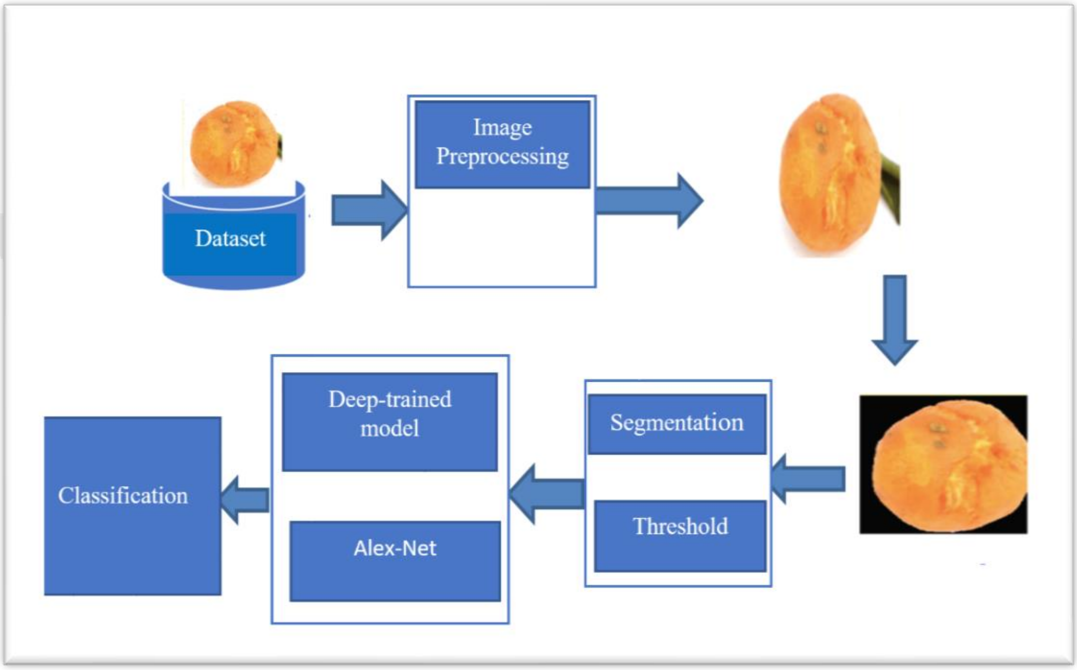

Figure 4.8: Architecture of the Classical Approach.

For preprocessing, the images are resized in size 224 * 224 by bilinear interpolation from [29]. The median filter being qualified to suppress pepper and salt noise, we applied the median filter to remove salt noise from the images. For the segmentation part, we chose as the zone of interest the part of the image containing the fruit. For this, we opted for a thresholding algorithm to separate the background and the foreground.

- Background subtraction

In DIP, an image is represented by matrix form. So, let the original image be denoted I and in RGB color space.

- From the grayscale image, we calculated the background mask by using a threshold.The threshold is found by using graythresh function which performs otsu's method.
- Performed the thresholding using the value found from step 1 to find the foreground mask.
- Find the largest connected component and perform noise-cleaning (imclose operation).And calculate the BG from the FG, and zero out the corresponding locations in the original input image.

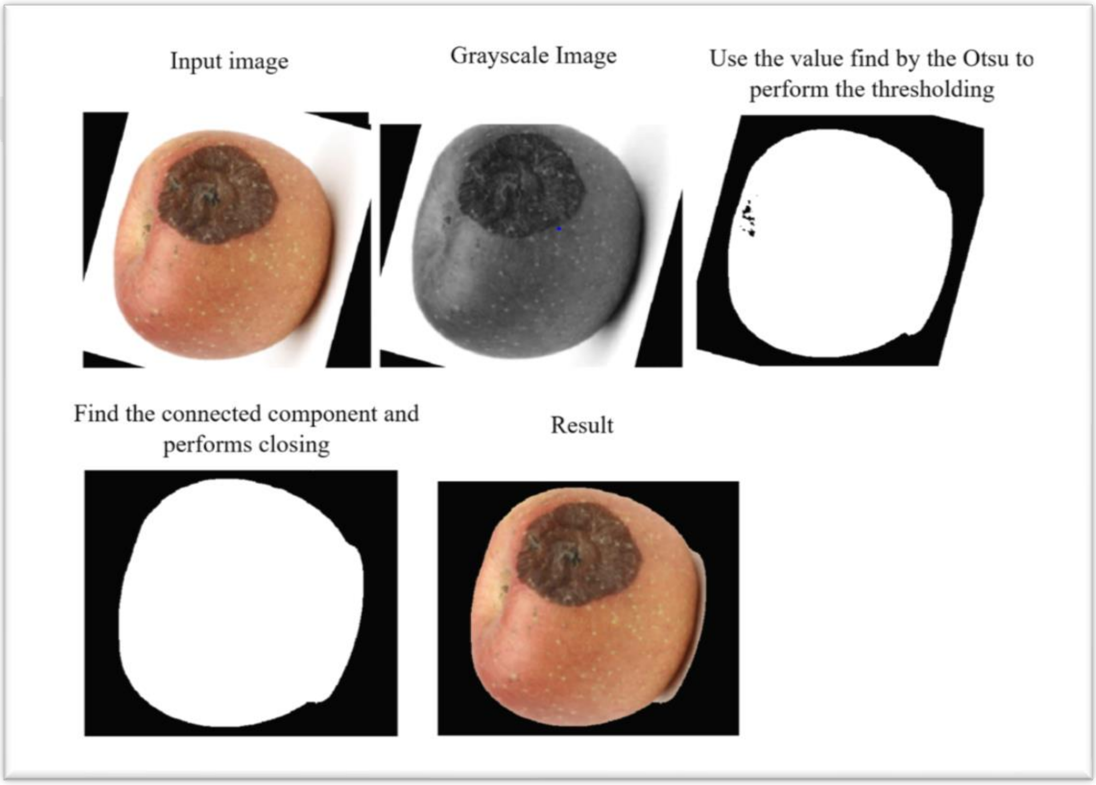

Figure 4.9: Pipeline of segmentation.

```
im = mat2gray(im(:,:,3));

%determines a threshold to distinguish between the leaf and its surroundings.
T = graythresh(im);

%defines FG as all the values below the threshold
%Also, keeps just the biggest connected component and perform noise
%reduction.
FG = im < T;
FG = bwareafilt(FG,1);
FG = imclose(FG,strel('disk',2));

%defines the background as the opposite of the foreground
BG = ~FG;
I(repmat(BG,1,1,3)) = 0;

%smooth the output
I(:,:,1) = medfilt2(I(:,:,1));
I(:,:,2) = medfilt2(I(:,:,2));
I(:,:,3) = medfilt2(I(:,:,3));
```

Figure 4.10: Example code segmentation.

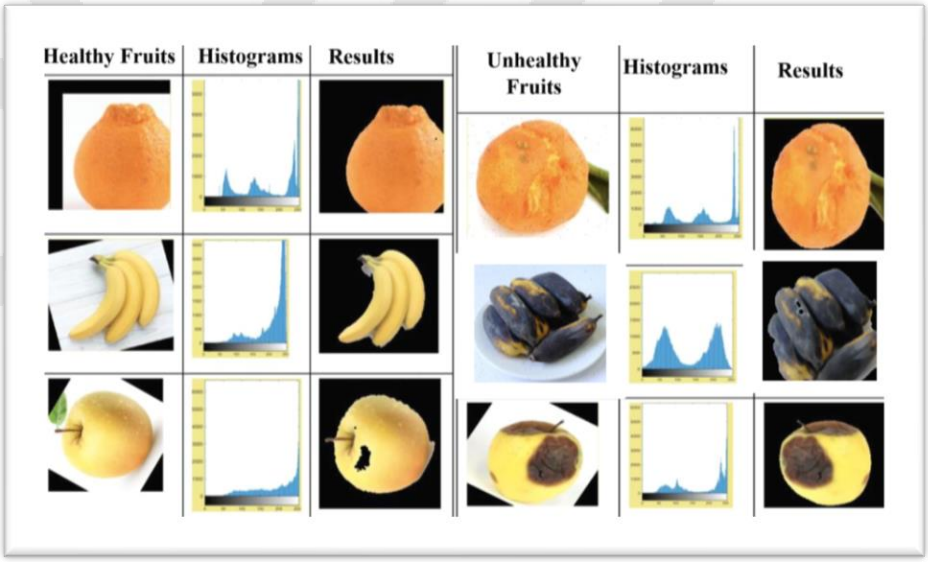

Figure 4.11: Example of segmented and resized images,in the right healthy fruits image, in the left rotten fruits images.

5. RESULTS AND EVALUATIONS

In this chapter, we present the results achieved with the methods discussed in section 4. As mentioned in the same section, three sets of experiments were performed. For each network, we presented the different results obtained from the conversion of the RGB color space to other color spaces as well as the fusion approach. The different tables illustrate the training and validation accuracies, but only the validation accuracy for every model will be used for the comparison and discussion parts. We have different number of epochs for each color space. firstly,we defined 1000 epochs, but with the concept of early stopping, it stops, if the accuracy is not improving for consecutive 10 epochs, this method assumes that, the accuracy will not increase thereby do not need to train any further. As we compare the performance of each color space, the maximum accuracy of each color space and the maximum accuracy of the fusion approach were used.

5.1. Experiment 1: Custom-CNN

Figure 5.1 shows the accuracy achieved by Custom-CNN. İn this section we looked into the effects of each color space and compared their performances.. We plot first the accuracy for different color spaces.From Table 5.1,we can observe that, the results obtained were highest for LAB color space with an accuracy of 89.78%.The RGB color space had the second-highest accuracy with 84.71%, while the lowest accuracy was obtained by HSV color space 74.65%.

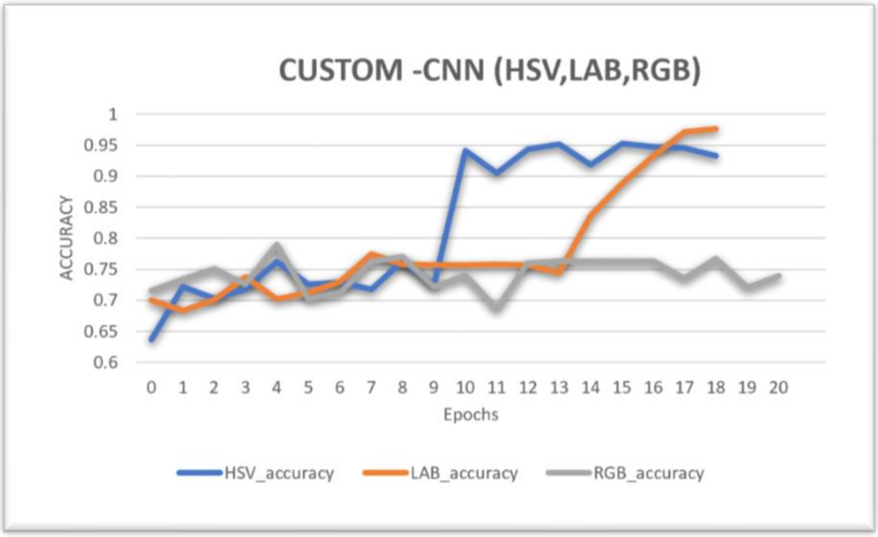

Figure 5.1: Validation Accuracy of custom-CNN on different color spaces.

Looking at the performance of the fusion approach, Normally the fusion of the color should create a new input for CNN. The idea behind the fusion of the color spaces was to obtain a high classification accuracy.From Table 5.1, we can see that, the fusion approach obtained the highest accuracy compared to all the single color spaces used with a validation accuracy of 91.23 %.

Table 5.1: Comparison of Validation Accuracy for single and combination of color spaces with Custom-CNN.

	Training-Acc(%)	Validation-acc(%)
RGB	76.71	84.71
HSV	95.25	74.65
LAB	99.66	**89.78**
RGB+HSV+LAB	97.93	**91.23**

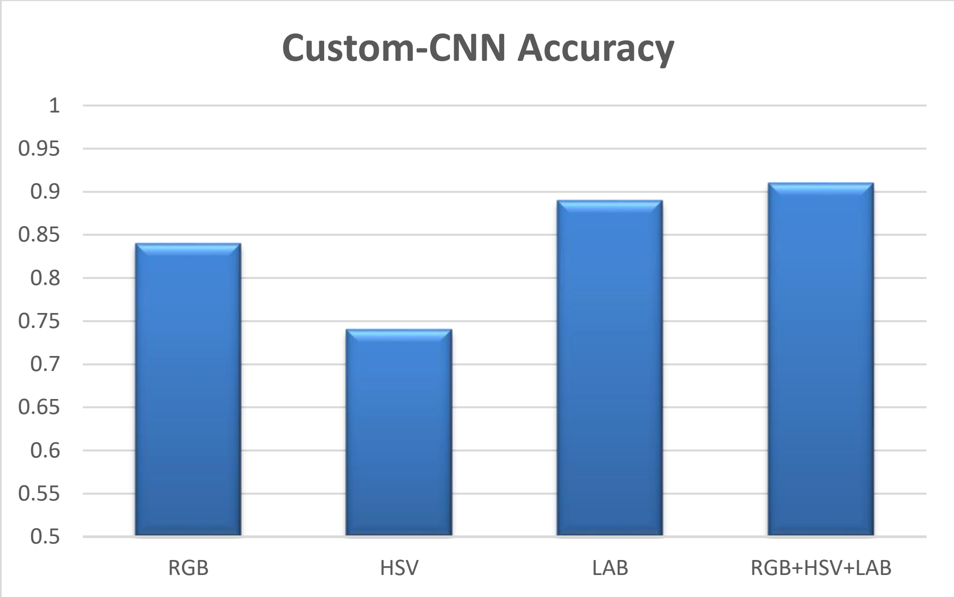

Figure 5.2: Validation Accuracy of Custom-CNN on fusion color spaces.

5.2. Experiment 2: Alex-Net

İn this section another technique well known in the field of research called Alex-Net was used on the same color spaces like in the Custom-CNN and results compared. The graph in Figure 5.3 shows the results in detail where, RGB color space achieved the highest accuracy among all the color spaces with 95.25%, followed by the LAB color space with 91.23%. And lastly, HSV performs the least with an accuracy of 84.71%.

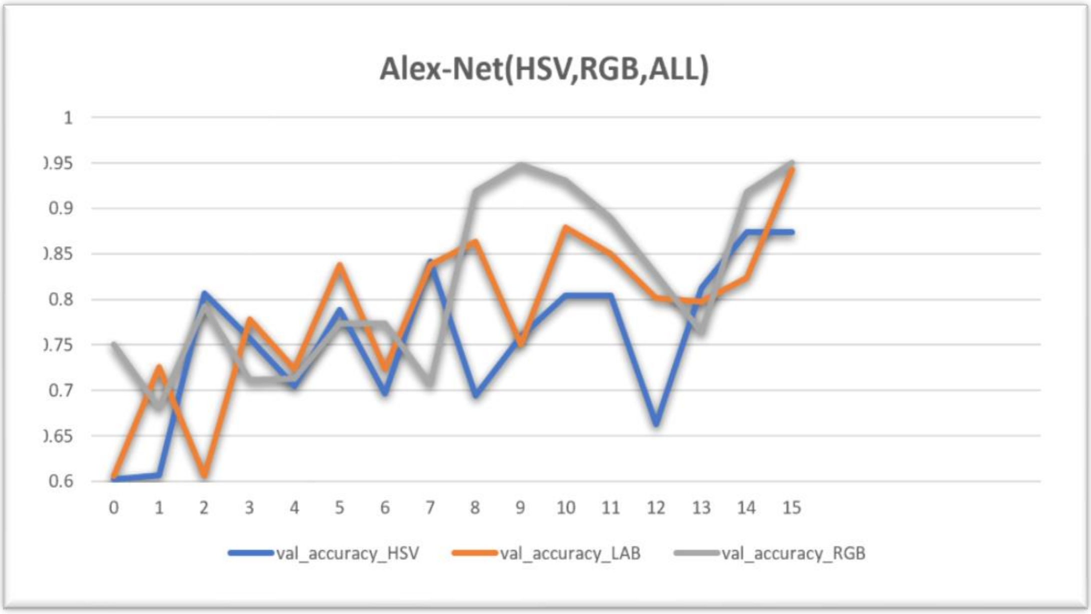

Figure 5.3: Validation Accuracy of Alex-Net on different color spaces.

Concerning the combination, the Alex-Net performs very well compared to the single color spaces with an accuracy of 97.94% as shown in Figure 5.4.

Table 5.2: Comparison of Validation Accuracy for single and combination of color spaces with Alex-net.

Color space	Training-Acc(%)	Validation-Acc(%)
RGB	97.13	**95.11**
HSV	95.32	84.71
LAB	97.76	92.68
RGB+HSV+LAB	99.73	**97.94**

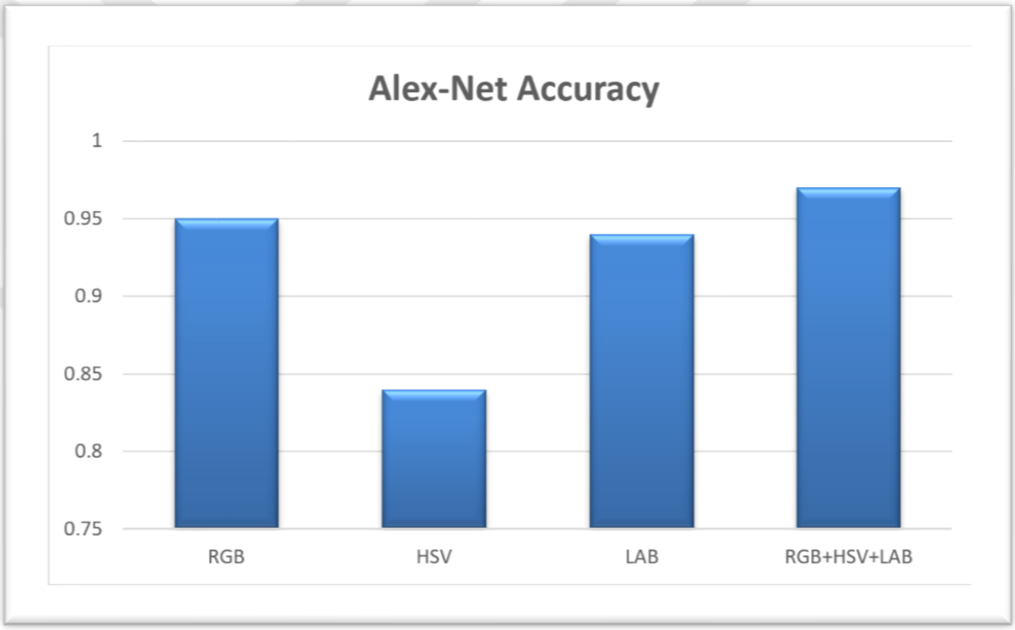

Figure 5.4: Validation Accuracy of Alex-Net on different color spaces.

Table 5.3 summarized the performance of the Alex-Net and Custom-CNN. It gives the accuracies of both networks with single color spaces as well as the fusion approach.

Table 5.3: Comparative table between Alex-Net and Custom-Cnn.

		Training (%)	**Validation (%)**
Alex-Net	RGB	97.13	**95.11**
	HSV	95.32	84.71
	LAB	97.76	92.68

	RGB+HSV+LAB	99.73	**97.94**
Custom-CNN	RGB	76.71	84.71
	HSV	95.25	74.65
	LAB	99.66	**89.78**
	RGB+HSV+LAB	97.93	**91.23**

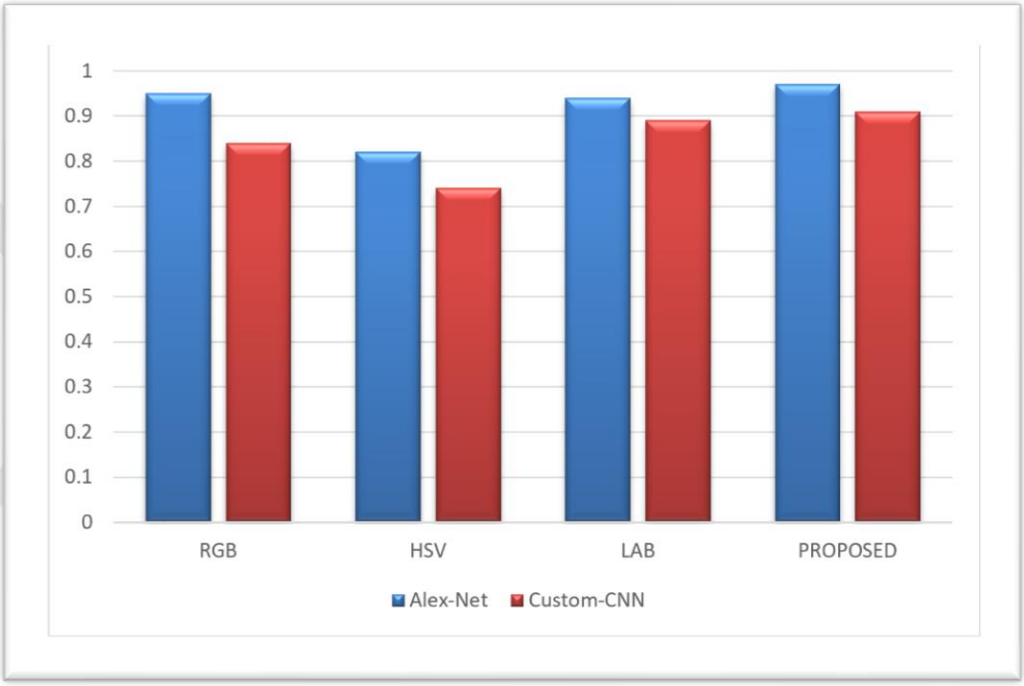

Figure 5.5: Comparison of Validation accuracy of Alex-Net and Custom-CNN.

5.3. Experiment 3: comparisons with Other Approaches

To complete our experiment, we compared our model with the different approaches discussed in section 2 and section 3 (transfer learning and the image processing approach). For experimental purposes three types of models were studied.The first approach been the classical image processing approach where we find the following steps:pre-processing, segmentation(Threshold), feature extraction (LBP) and the last being the classification(SVM) step. The second approach being the learning transfer.For the transfer learning approach we choose Alex network as feature generator, after we pre-processed and segmented the images as explained in section

4.And the third approach is our fusion approach.the fusion approach with Alex-Net got the highest compared to the other methods with an accuracy of 97.94%. Table 5.4 shows more of the details.

Table 5.4: Comparison of the proposed method and Other Approaches.

	Classical approach	Transfer-Learning	Proposed	Proposed
	SVM+LBP	Alex-Net (Deep features)	Fusion (Alex-Net)	Fusion (Custom-CNN)
	71.60	84.20	97.94%	91.23

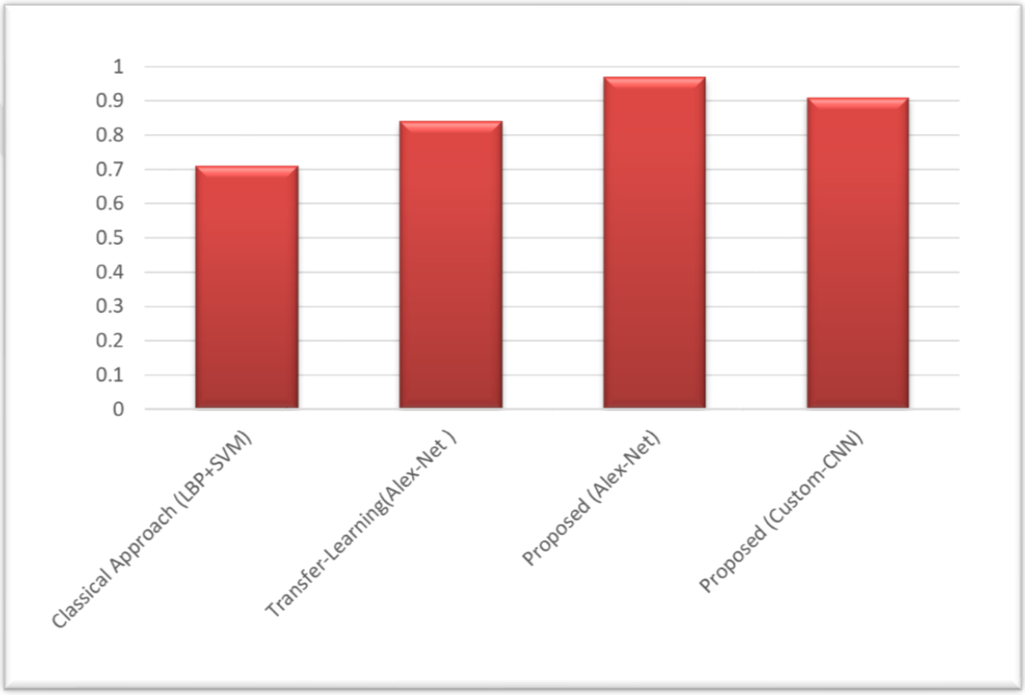

Figure 5.6: Comparison of Validation accuracy of the proposed method and Other Approaches.

5.4. Discussions

Referring to the results obtained, as shown in Figure 5.5, and 5.6, Table 5.3, a general conclusion can be drawn on the performance of classifier.The results of individual color spaces, are given in Figure 5.5 and shows that the color spaces have an impact on the accuracy. The different results of the Alex-net and the Custom-CNN were given. By training separately the color space, the LAB color space holds the highest accuracy for the Custom-CNN and the RGB color space for the Alex-Net with

89.78% and 95.11% respectively. The HSV color space performed the least on both networks with an accuracy of 74.65% for the Custom-Cnn and 84.71% for the Alex-net. However the RGB color space performed relatively close to the LAB color space.

From the obtained results in Figure 5.5, we can observe the different results achieved by our fusion approach. From these results, it can be generalized that the fusion approach performs better than all the single color spaces were the highest accuracy was accomplished by the Alex-Net. The accuracy with the concatenation in the Custom-CNN was 91.23% and on the Alex-net, 97.94%. That's means our network performed well despite the fact that, it is less complex than Alex-Net, however Alex-Net obtained a better result with a difference of 6% compared to the custom-CNN.

Referring to the results obtained in Figure 5.6, where different approaches have been tested on the same dataset. Our fusion approaches with Alex-Net model and the Custom-CNN performed very well compared to the image processing approach and the transfer learning approach. However, for training such a model (with multiple colors) whether with a complex model or not, many questions arise like, the time needed for multiple color space conversion as this causes overheads. However, the importance of the fusion model is that, the model can give us a higher accuracy. Additionally, it was determined that Alex-Net consistently outperformed Custom-CNN architecture.The accuracy with the concatenated color spaces from the Alex-Net was 97.94% and the accuracy of Custom-Cnn was 91.23% against 71.60% for the image processing approach and 84.20% for the transfer learning with Alex-Net.These results lead us to conclude that the fusion of these color spaces can increase the performance of a model.

6. CONCLUSION AND PERSPECTIVES

In this last chapter, we summarize the important elements of this research into three different sections namely, the overview of our research, contribution of research and the limitations of the proposed approaches.

6.1. Overview of the Research

This section is divided in to five parts, the first part started by introducing our topic in detail as well as our vision of the work to be carried out. The second part, looked in to the state of art regarding the inspection and quality control of fruits and vegetables in agriculture were researchers conducted similar study on the inventory of detection and classification methods used for such problems. Thirdly, we presented our approach to solve the food classification problem then, details of the implementation are presented in the fourth part, finally the obtained results are discussed and compared in the fifth part.

6.2. Contribution of research

The main objectives of this thesis were to classify fruit diseases by examining the influence of different color spaces on the performance of classification with convolutional neural network. To do this, 3 color spaces in the name, RGB, LAB, and HSV were choosen. We futher went on to introduce the concatenated form of these color spaces in order to increase the accuracy of fruit disease classification. İn performing these experiments, two different datasets were used, one of which was taken from kaggle called fruits fresh and rotten for classification and a new dataset that we gathered ourselves consisting of only apples in both it’s rotten and fresh forms with a propotion of 126 rotten and 540 fresh apples. Two CNN models were then used as well and those models consider not only a single color space,but multiple color spaces as inputs, we designed one CNN model called Custom-CNN and the traditional Alex-Net model. The accuracy obtained from the concatenation approach of the custom-CNN was 91.23% and 97.94% for Alex-Net. With regards to the classical approaches, the performance obtained were 84.20% for transfer learning approach and 71.60% for the classical image processing approach, concerning the single color space, the highest accuracy was 89.78% for the Custom-CNN with LAB color space and 95.11% for Alex-Net with the RGB color space. The HSV color space performed the least on both networks with an accuracy of 74.65% for the Custom-CNN and 84.71% for the Alex-Net.

6.3. Discussion on the limits of the proposed approaches

Although this research comes along with a practical contribution, the results and the contributions formulated at the end of this study were however limited by certain constraints like the impossibility of collecting important disease dataset, as we earlier discussed in the second chapter, there are several diseases that can affect fruits and vegetables but as part of our research, we focused solely on classifying infected fruits and did not test the approach on a large set of data containing fruits with several diseases. This limitation is due to the fact that accessing a database with such datasets is difficult or sometimes impossible. The second limitation we realised was, memory limitation, the training takes too much memory space and the network took over 3 days for the execution of just one model.

REFERENCES

[1] Bhange M., Hingoliwala H. A., (2015), "Smart farming: Pomegranate disease detection using image processing", Procedia Computer Science, 58, 280-288.

[2] Husin Z. B., Shakaff A. Y. B. M., Aziz A. H. B. A., Farook R. B. S. M., (2012), "Feasibility study on plant chili disease detection using image processing techniques". In 2012 Third International Conference on Intelligent Systems Modelling and Simulation, 291-296.

[3] Marfo R., (2016), "Measurıng The Severıty Of Fungı Caused Dısease On Leaf Usıng Trıangular Thresholdıng Method", Master Thesis, University of Science and Technology.

[4] Shaikh R. P., Dhole S. A., (2017), "Citrus leaf unhealthy region detection by using image processing technique". In 2017 International conference of Electronics, Communication and Aerospace Technology (ICECA), 420-423.

[5] Kim M. S., Lefcourt A. M., Chen Y. R., Tao Y., (2005). "Automated detection of fecal contamination of apples based on multispectral fluorescence image fusion". Journal of food engineering, 85-91.

[6] Habib M. T., Majumder A., Jakaria A. Z. M., Akter M., Uddin M. S., Ahmed F., (2018), "Machine vision-based papaya disease recognition". Journal of King Saud University-Computer and Information Sciences, 1-31.

[7] Li X., Zhu W., (2011)," Apple grading method based on features fusion of size, shape and color", Procedia Engineering, 15, 2885-2891.

[8] Dubey S. R., Jalal A. S., (2012)," Detection and classification of apple fruit diseases using complete local binary patterns", IEEE 346-351.

[9] Khan M. A., Akram T., Sharif M., Awais M., Javed,K., Ali H., Saba T., (2018),"CCDF: Automatic system for segmentation and recognition of fruit crops diseases based on correlation coefficient and deep CNN features",Computers and electronics in agriculture, 155, 220-236.

[10] Capizzi G., Sciuto G. L., Napoli C., Tramontana E., Woźniak M., (2015), "Automatic classification of fruit defects based on co-occurrence matrix and neural networks", IEEE, 861-867.

[11] Unay D., Gosselin B., Kleynen O., Leemans V., Destain M. F., Debeir, O., (2011), "Automatic grading of Bi-colored apples by multispectral machine vision", Computers and electronics in agriculture, 75, 204-212.

[12] Sharif M., Khan M. A., Iqbal Z., Azam M. F., Lali M. I. U., Javed M. Y., (2018), "Detection and classification of citrus diseases in agriculture based on

optimized weighted segmentation and feature selection", Computers and electronics in agriculture, 150, 220-234.

[13] Abdelhamid D., Salah R., Chaouki B. M., Abdelmalik T. A., (2010), “Classification des images des dattes par SVM: contribution à l’amélioration du processus de tri” , 159–167.

[14] Bhargava A., Bansal A., (2018)," Fruits and vegetables quality evaluation using computer vision", A review, Journal of King Saud University-Computer and Information Sciences, 1-15.

[15] Hong T., Tsamis D., (2006), "Use of KNN for the Netflix prize" CS229 Projects. pdf, 1-5.

[16] Wallelign S., (2017), “Soybean Plant Disease Identification Using Convolutional Neural Network”, 146–151.

[17] Lamani S.B., (2018), “Pomegranate Fruits Disease Classification with K Means Clustering”, 3, 74–80.

[18] Mahendran R., Gc J., Alagusundaram K., (2015), “Application of Computer Vision Technique on Sorting and Grading of Fruits and Vegetables” , J. Food Process. Technol, 10, 2157-7110.

[19] Mccaig C. D., (1990), “Electric Fields in Vertebrate Repair ”. 310. (Alan R. Liss, New York, 1989.) $69.50 hardback. ISBN 0 8451 4274 ”, Exp. Physiol, no. 2, 280–281.

[20] Schmidhuber J., (2015) “Deep Learning in neural networks: An overview”, Neural Networks, 85–117.

[21] Bengio Y., Courville A., Vincent P., (2013), “Representation learning: a review and new perspectives”, IEEE transactions on pattern analysis and machine intelligence, 8,1798–828.

[22] Bengio Y. (2009), "Learning deep architectures for AI. Foundations and trends® in Machine Learning", 2, 1-127.

[23] Randrianarivony M. I ., (2018) , “Detection de concepts et annotation automatique d’images medicales par apprentissage profond” , Master Thesis, University of Antananarıvo1, 1-71.

[24] Amara J., Bouaziz B., Algergawy A., (2017), “A Deep Learning-based Approach for Banana Leaf Diseases Classification” ,79–88.

[25] Mohanty S. P., Hughes D. P., Salathé M., (2016),“Using Deep Learning for Image-Based Plant Disease Detection” , Front. Plant Sci, 7, 1419.

[26] Yosinski J., Clune J., Bengio Y., Lipson H., (2014), “How transferable are features in deep neural networks?”, 1–9.

[27] Türkoğlu M., Hanbay D., (2019), "Plant disease and pest detection using deep learning-based features," Online.Journals.Tubitak.Gov.Tr, 1636–1651.

[28] Shankar S., Garg V. K., Cipolla R., (2015),"DEEP-CARVING: Discovering visual attributes by carving deep neural nets," Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit, 3403–3412.

[29] Gonzalez R. C., Woods R. E., Masters B. R.,(2009), "Digital Image Processing", Third Edition, vol. 14, 2.

[30] Kang M. G., Katsaggelos A. K, (1995), "General choice of the regularization functional in regularized image restoration", IEEE Transactions on Image Processing, 594-602.

[31] McKinley G. A., (1990),"An introduction to digital image processing," Clin. Microbiol. Newsl, 89–92.

[32] Rusk N. (2015), "Deep learning," Nat. Methods, 1-35.

BIOGRAPHY

Rhudie Grace ZANG EDZANG was borns in Gabon in the province of Ogooué-Maritime.In 2016, she successfully completed the Bachelors of computer science in the Faculty of Engineering at the institute of technology of Libreville (Gabon) and started her graduate education at Gebze Technical University, Institute of Science and computer Engineering in the next year.

APPENDICES

Appendix A : Sample the predictions with some images (Transfer learning).

Table A1.1: Prediction with some rotten fruits.

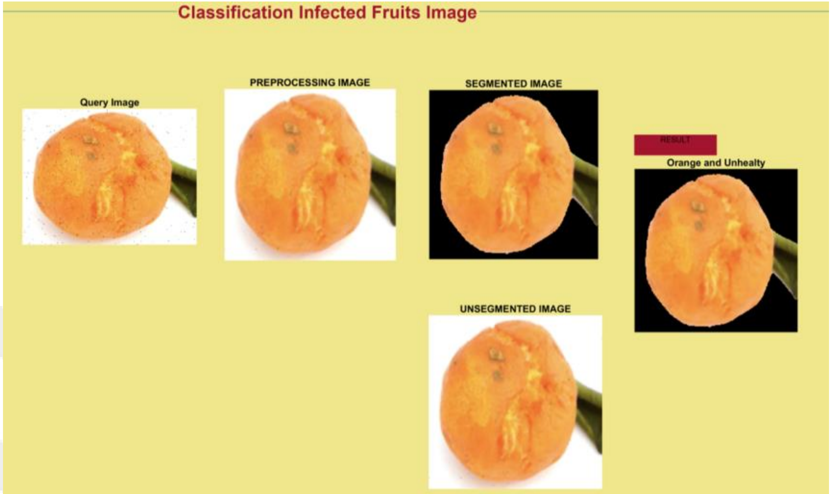

Table A1.1: Continued.

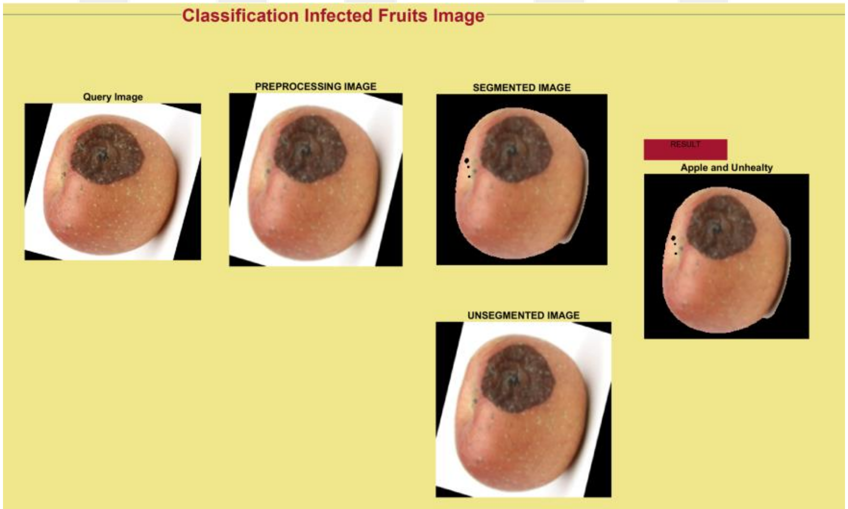

Table A1.1: Contiued.